The Role of Prices in Peer-Assisted Content Distribution

Christina Aperjis*, Michael J. Freedman†, and Ramesh Johari*
*Stanford University, †Princeton University

Abstract

Peer-assisted content distribution matches user demand for content with available supply at other peers in the network. Inspired by this supply-and-demand interpretation of the nature of content sharing, we employ *price theory* to study peer-assisted content distribution. In this approach, the market-clearing prices are those which exactly align supply and demand, and the system is studied through the characterization of price equilibria.

Our work provides two separate steps forward. First, we rigorously analyze the efficiency and robustness gains that are enabled by price-based multilateral exchange. We show that multilateral exchanges satisfy several desirable efficiency and robustness properties that bilateral exchanges such as BitTorrent do not, particularly when considering multiple files. Second, we propose and evaluate a system design that realizes many of the benefits of a price-based multilateral exchange; our design encourages sharing of desirable content and network-friendly resource utilization.

Bilateral barter-based systems such as BitTorrent have been attractive in large part because of their simplicity; however, little attention has been devoted to studying the efficiency and robustness lost in return for this simplicity. Our research takes a significant step in filling this gap, both through formal analysis and system design.

1 Introduction

Peer-to-peer systems have been wildly successful as a disruptive technology for content distribution. Varying accounts place peer-to-peer (P2P) traffic as comprising anywhere between 35% and 90% of "all" Internet traffic, with BitTorrent accounting for its large majority [3]. Perhaps BitTorrent's biggest technical contribution is its content-exchange mechanisms—rate-based tit-for-tat [6], or *bilateral barter*—that users widely view as incentivizing uploads.

While BitTorrent's usage numbers are certainly impressive, there are some fundamental problems with its resource allocation and incentive mechanisms, even beyond potential "free-riding" attacks already observed [13, 17, 26, 22]. Namely, the system can only perform bilateral barter by matching up well-suited pairs of nodes that have disjoint subsets of a file (or, more generally, files). Yet the discovery of stable peering relationships is slow in practice—measured in the tens of minutes [5] or, at least for high-bandwidth peers, requiring a linear brute-force search of other participants to find similar reciprocation rates [22]—if such reciprocation exists at all. In the end, altruistic uploading often turns out to be critical for providing continued content availability [10].

From an economics perspective, many of these problems ultimately have to do with "market failure" in the system: it's hard to find good reciprocation with bilateral barter alone. But economics also offers an alternative—*market-based multilateral exchange*—where the system matches user demand for content to available supply at other peers in the network. Multilateral exchange becomes especially efficient and incentive-compatible when combined with currency: Peers accrue revenue over time by uploading content, and they spend that revenue to download content. Given the potential for currency-backed exchange, we turn to *price theory* to study peer-assisted content distribution. Prices can help identify which files are "most useful" to disseminate, where resource congestion is occurring, and who is providing useful content to the system.

Our paper makes two main contributions. First, we provide a *rigorous theoretical analysis of price-based exchanges*, providing a foundation to compare the efficiency and robustness of the resource allocation achieved by these schemes. We show how one can model peer-to-peer file-sharing systems—such as BitTorrent and variants like BitTyrant [22]—in terms of *exchange ratios*, then use these abstractions to show that a multilateral price-based exchange scheme satisfies a number of desirable properties lacking in bilateral exchange, *e.g.*, equilibria in bilateral exchange may fail to exist, be inefficient if they do exist, and fail to remain robust to collusive deviations even if they are efficient. We analyze a range of pricing schemes, and conclude that simply maintaining a single price per peer suffices to achieve the benefits of price-based multilateral exchange. We also demonstrate the impact of currency on system dynamics. These results help clarify the tradeoffs inherent in choosing between bilateral and multilateral exchanges: simplicity in the former, and efficiency and robustness gains in the latter.

Our second contribution is a *system design—PACE (Price-Assisted Content Exchange)—that effectively and practically realizes multilateral exchange*. Its centerpiece is a market-based mechanism for exchanging currency for desired content, with a single price per peer as advocated by our theoretical analysis. The system fully specifies the algorithmic buy-and-sell behavior of users' clients: Honest users are completely shielded from any notion of prices, budgeting, allocation, or other market issues, yet strategic or malicious clients cannot unduly damage the system's efficient operation. Still, users are incentivized to contribute resources, as each user's performance is affected by his (hidden) budget. PACE introduces a hierarchical network model that captures the congestion points in a network for resource pricing, yet also supports the ability for ISPs to express preferences for

long-haul traffic carriage. This model ensures that P2P traffic has a strong incentive to remain local when possible, or at least traverse wide-area links that yield more efficient network usage—a strong form of *network friendliness*. Through simulations we validate that our system design provides incentives for participation and efficient network usage.

Finally, we complete PACE's design by proposing cryptographic protocols that ensure the security and ε-fairness of peer exchanges, and we discuss how to scale and even federate PACE's rendezvous and currency services, which provide discovery and banking for *collections* of files. Much like an ecosystem of tracker sites has arisen around BitTorrent's *logically*-centralized trackers [21], we envision a similar ecosystem around PACE's components as well.

The paper is organized as follows. Section §2 presents theoretical comparisons of bilateral and multilateral exchange, as well as various pricing schemes. Section §3 discusses the user incentives provided by our exchange model, while §4 presents our network model. Section §5 details the algorithmic mechanisms at the heart of our buy and sell clients, while §6 describes the associated system services and security mechanisms. Section §7 evaluates the resulting system in simulation, §8 discusses related work, and §9 concludes.

2 Prices in Peer-to-Peer Systems

This section provides a formal comparison of P2P system designs with bilateral barter, such as BitTorrent, and a market-based exchange of content enabled by a price mechanism to match supply and demand.

We start in Section §2.1 with a fundamental abstraction of content exchange in systems like BitTorrent: *exchange ratios*. The exchange ratio from one peer to another gives the download rate received per unit upload rate. We show that the rates achieved by existing protocols, such as BitTorrent and BitTyrant [22], can be naturally modeled through exchange ratios.

Exchange ratios are a useful formal tool because they directly allow us to compare bilateral P2P systems with price-based P2P systems. We carry out such a comparison in §2.2. We compare bilateral and multilateral P2P systems through the allocations that arise at equilibria. In particular, we show that bilateral equilibria may fail to exist, may be inefficient if they do exist, and correspond to multilateral equilibria if and only if they are robust to deviations by coalitions of users. These results provide formal justification of the efficiency and robustness benefits of multilateral equilibria.

We provide a comparative study of a range of price-based multilateral exchange mechanisms in §2.3. We compare systems that set one price per peer, one price per file, and one price per file per peer. We show that a system with one price per peer suffices to achieve the benefits of a price-based multilateral exchange. We conclude with a discussion of system dynamics in §2.4.

2.1 Exchange Ratios in Bilateral Protocols

Many P2P protocols enable exchange on a *bilateral* basis between peers: a peer i uploads to a peer j if and only if peer j uploads to peer i in return. Of course, such an exchange is only possible if each peer has something the other wants. The foremost examples of such a protocol are BitTorrent and its variants. While such protocols are traditionally studied solely through the rates that peers obtain, in this section we provide an interpretation of these protocols through *exchange ratios*. As exchange ratios can be interpreted in terms of prices, these ratios will allow us to compare bilateral barter-based P2P systems with multilateral price-based P2P systems in the following section.

Let r_{ij} denote the rate sent from peer i to peer j in an instantiation of a BitTorrent swarm. We define the *exchange ratio* between peer i and peer j as the ratio $\gamma_{ij} = r_{ji}/r_{ij}$; this is the download rate received by i from j, per unit of rate uploaded to j. By definition, $\gamma_{ij} = 1/\gamma_{ji}$. Clearly, a rational peer i would prefer to download from peers with which he has higher exchange ratios.

The exchange ratio has a natural interpretation in terms of prices. An equivalent story emerges if we assume that peers charge each other for content in a common monetary unit, but that all transactions are *settlement-free*, *i.e.*, no money ever changes hands. In this case, if peer i charged peer j a price p_{ij} per unit rate, the exchange of content between peers i and j must satisfy:

$$p_{ij} r_{ij} = p_{ji} r_{ji}$$

We refer to p_{ij} as the *bilateral price* from i to j. Note that the preceding condition thus shows the exchange ratio is equivalent to the ratio of bilateral prices: $\gamma_{ij} = p_{ij}/p_{ji}$ (as long as the prices and rates are nonzero).

What is the exchange ratio for BitTorrent? A peer splits its upload capacity equally among those peers in its active set from which it gets the highest download rates. Let α be the size of the active set. Suppose all rates r_{kj} that peer j receives from peers $k \neq i$ are fixed and let R_j^α be the α-th highest rate that j receives. Let B_j be the upload capacity of peer j. Then, r_{ji} depends on r_{ij}. In particular,

$$r_{ji} = \begin{cases} B_j/\alpha & \text{if } r_{ij} > R_j^\alpha \\ 0 & \text{otherwise} \end{cases}$$

Thus for BitTorrent, the exchange ratio is $\gamma_{ij} = B_j/(\alpha \cdot r_{ij})$ if peer i is in the active set, and zero otherwise. Note that the exchange ratios $\gamma_{i_1,j}$ and $\gamma_{i_2,j}$ may be different for two peers i_1, i_2 in j's active set.

The exchange ratio γ_{ij} *decreases* with r_{ij} as long as peer i is in peer j's active set (in which case r_{ji} is constant). Hence, a strategic peer i would prefer to choose r_{ij} as small as possible while remaining in j's active set. This behavior is exactly the approach taken by the BitTyrant [22] variation on BitTorrent. In fact, if all peers follow this policy, then $r_{ij} = R_j^\alpha$ for all peers i in j's active set. Note that in this case, $\gamma_{ij} = B_j/(\alpha \cdot R_i^\alpha)$. Thus, peer j has the same exchange ratio to all peers i with which he bilaterally exchanges content.

Bilateral Peer Optimization:

maximize $V_i(\mathsf{d}_i)$
subject to $d_{if} = \sum_j r_{jif}$ for all f
$r_{ijf} = 0$ if $f \notin F_i$
$\sum_{j,f} r_{ijf} \leq B_i$
$\sum_f r_{jif} = \gamma_{ij} \sum_f r_{ijf}$ for all j
$\mathsf{r} \geq 0$.

Multilateral Peer Optimization:

maximize $V_i(\mathsf{d}_i)$
subject to $d_{if} = \sum_j r_{jif}$ for all $f \in T_i$
$r_{ijf} = 0$ if $f \notin F_i$
$\sum_{j,f} r_{ijf} \leq B_i$
$\sum_{j,f} p_j r_{jif} = p_i \sum_{j,f} r_{ijf}$
$\mathsf{r} \geq 0$.

Figure 1: **Optimization problems for price-based exchange.**

The preceding discussion highlights the fact that the rates in a bilateral P2P system can be interpreted via exchange ratios. Thus far we have assumed that *transfer rates* are given, and exchange ratios are computed from these rates. In the next section, we turn this relationship around: we explicitly consider an abstraction of bilateral P2P systems where peers react to given exchange ratios, and compare the resulting outcomes to price-based multilateral exchange.

2.2 Bilateral vs. Multilateral Exchange

Motivated by the discussion in the preceding section, this section rigorously analyzes the efficiency properties of price-based bilateral and multilateral mechanisms. Peers explicitly react to prices, and we compare the schemes through their resulting price equilibria. All proofs for this section can be found in the paper's appendix.

In the formal model we consider, a set of peers N shares a set of files F. Peer i has a subset of the files $F_i \subseteq F$, and is interested in downloading files in $T_i \subseteq F - F_i$. Throughout, we use r_{ijf} to denote the rate at which user i uploads file f to user j. We then let $d_{if} = \sum_j r_{jif}$ be the rate at which user i downloads file f, respectively. We use sans serif to denote vectors, *e.g.*, $\mathsf{d}_i = (d_{if}, f \in T_i)$ is the vector of download rates for user i.

We measure the desirability of a download vector to peer i by a *utility function* $V_i(\mathsf{d}_i)$ that is nondecreasing in every d_{if} for $f \in T_i$. We temporarily ignore any resource constraints within the network; we assume that transfers are only constrained by the upload capacities of peers. The upload capacity of peer i is denoted B_i.

We start by considering peers' behavior in bilateral schemes, given a vector of exchange ratios (γ_{ij}). Peer i solves the bilateral optimization problem given in Figure 1. Note that we allow peers to bilaterally exchange content over multiple files, even though this is not typically supported by swarming systems like BitTorrent.

By contrast, in a *multilateral price-based exchange*, the system maintains one price per peer, and peers optimize with respect to these prices. In a slight abuse of notation, we denote the price of a peer i by p_i. Figure 1 also gives the peer optimization problem in multilateral price-based exchange. Note that the first three constraints (giving download rates, ensuring peers only upload files they possess, and meeting the bandwidth constraint) are identical to the bilateral peer optimization. While the bilateral exchange implicitly requires peer i to download only from those peers to whom he uploads, no such constraint is imposed on multilateral exchanges: peer i accrues capital for uploading, and he can spend this capital however he wishes for downloading.

For bilateral (resp., multilateral) exchange, an *equilibrium* is a combination of a rate allocation vector and an exchange ratio vector (resp., price vector) such that all peers have solved their corresponding optimization problems. In this case, the exchange ratios (resp., prices) have exactly aligned supply and demand: for any i, j, f, the transfer rate r_{ijf} is simultaneously an optimal choice for both the uploader i and downloader j.

Definition 1 *The rate allocation* r^* *and the exchange ratios* $(\gamma^*_{ij}, i, j \in N)$ *with* $\gamma^*_{ij} > 0$ *for all* $i, j \in N$ *constitute a* ***bilateral equilibrium*** *if for each peer* i, r^* *solves the Bilateral Peer Optimization problem given exchange ratios* $(\gamma^*_{ij}, j \in N)$.

Definition 2 *The rate allocation* r^* *and the peer prices* $(p^*_i, i \in N)$ *with* $p^*_i > 0$ *for all* $i \in N$ *constitute a* ***multilateral equilibrium*** *if for each peer* i, r^* *solves the Multilateral Peer Optimization problem given prices* $(p^*_j, j \in N)$.

This latter definition is the traditional notion of competitive equilibrium in economics [19]. A multilateral equilibrium can be shown to exist under general conditions in our setting [2]. Moreover, the corresponding allocation is Pareto efficient, *i.e.*, there is no way to increase the utility of some peer without decreasing the utility of some other peer. A bilateral equilibrium, on the other hand, does not always exist, and, even when it exists, the allocation may not be efficient.

Example 1 *Consider a system with n peers and n files, for* $n > 2$. *Each peer* i *has file* f_i *and wants* $f_{(i \bmod n)+1}$. *With these utilities, no bilateral exchange can satisfy all peers, and a bilateral equilibrium does not exist.*

This observation is not particularly surprising: after all, exchange is far more restricted in a bilateral equilibrium than in a multilateral equilibrium. We focus instead on determining conditions under which a bilateral equilibrium yields a multilateral equilibrium. The following is a key step in establishing the relationship between bilateral and multilateral equilibrium.

Proposition 2 *Consider a bilateral equilibrium with exchange ratios* γ_{ij} *for every pair of peers* i, j. *If there exist prices* p_i *for all* $i \in N$ *such that* $\gamma_{ij} = p_i / p_j$ *for all* $i, j \in N$, *then the bilateral equilibrium allocation is also a multilateral equilibrium allocation.*

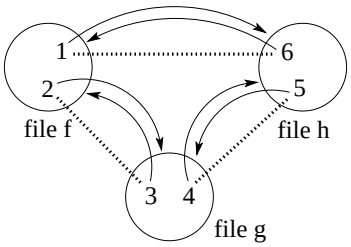

Figure 2: **Bilateral equilibrium for Ex.3. Peers $\{1,2\}$ have file f, peers $\{3,4\}$ have file g and peers $\{5,6\}$ have file h. Solid arrows are drawn from a peer to its desired file, *e.g.*, 1 and 4 want h. Heavy dotted lines are drawn between peers that barter at the unique bilateral equilibrium.**

The proof shows that the constraints in the Bilateral Peer Optimization problem are equivalent to the constraints in the Multilateral Peer Optimization problem when $\gamma_{ij} = p_i/p_j$ for all $i, j \in N$. This proposition is quite revealing: it shows that if exchange ratios are "fair," in the sense that they yield a unique price per peer, then the bilateral equilibrium allocation is also a multilateral equilibrium allocation.

We have already seen that a bilateral equilibrium need not exist, whereas multilateral equilibria always exist; thus, the two concepts are not equivalent (in general). We now show that even if a bilateral equilibrium exists, it does not necessarily yield one price per peer, and thus is not always equivalent to a multilateral equilibrium.

Example 3 *There are 6 peers ($\{1,2,3,4,5,6\}$) and 3 files ($\{f,g,h\}$) in the system, with file allocation and demand as shown in Figure 2. The upload capacities of peers are $B_1 = 2$, $B_j = 1, \forall j \neq 1$. At the unique bilateral equilibrium, the following pairs exchange: $\{1,6\}$, $\{2,3\}$ and $\{4,5\}$. Thus for these pairs, the equilibrium exchange ratios must be $\gamma_{16} = 1/2$, $\gamma_{32} = 1$ and $\gamma_{54} = 1$.*

The optimality conditions in the bilateral equilibrium must ensure that peer 1 does not wish to download from peer 5 instead of peer 6, which implies that we must have $\gamma_{15} \leq \gamma_{16} = 1/2$. Similarly, we must have $\gamma_{53} \leq \gamma_{54} = 1$, and $\gamma_{31} \leq \gamma_{32} = 1$. Note that we thus have $\gamma_{15}\gamma_{53}\gamma_{31} \leq 1/2$. However, this implies there do not exist prices per peer p_i such that $\gamma_{ij} = p_i/p_j$, since such a price vector would imply $\gamma_{15}\gamma_{53}\gamma_{31} = 1$. Thus the bilateral equilibrium cannot be a multilateral equilibrium.

The problem in this example is that given the exchange ratios, peers $\{1,3,5\}$ can benefit by deviating together. If this set chooses upload rates $r'_{13} = 1/3, r'_{35} = 1/4, r'_{51} = 1/5$, while reducing upload rates to their original trading partners accordingly, then each peer in $\{1,3,5\}$ obtains a resulting download rate strictly larger than 1 (the download rate each of these peers gets at the bilateral equilibrium). For example, user 1 obtains a total download rate of $1/5$ (from user 5) plus $5/6$ (from user 6, who in turn gets $r'_{16} = 5/3$), which results in a rate greater than 1.

Inspired by this observation, we show next that if a bilateral equilibrium satisfies an additional robustness to joint deviation by a coalition of peers, and if each peer is only interested in one file, then it must be a multilateral equilibrium. We formalize this result using the notion of the *core* [19]. An allocation has the core property with respect to given exchange ratios if no coalition of peers can strictly improve the utility of all its members by bartering with peers outside the coalition, subject to the given exchange ratios. Inside the coalition, peers do not need to follow the exchange ratios, and they may allocate rates in any way subject to bandwidth constraints.

Definition 3 *Given exchange ratios $(\gamma_{ij}, i, j \in N)$, an allocation* r *is* feasible *for a set of peers S with respect to γ if:*

(i) $r_{ijf} = 0$ *if* $f \notin F_i$*;*

(ii) $\sum_{j,f} r_{ijf} \leq B_i$ *for all* $i \in S$*;*

(iii) $\sum_{j \notin S} \sum_f r_{jif} = \sum_{j \notin S} \gamma_{ij} \sum_f r_{ijf}$*, for all* $i \in S$*.*

The first condition ensures that all peers only upload files they have. The second condition ensures that all peers in S do not exceed their upload constraints. The third ensures that any exchanges from peers in S to peers outside S take place at the given exchange ratios. Both bilateral and multilateral equilibrium allocations are trivially feasible for any set of peers with respect to the equilibrium exchange ratios.

Definition 4 *Given fixed exchange ratios for every pair of files, a coalition S* blocks *a feasible allocation* r^* *if there exists a feasible allocation* r *for S such that $V_i(\sum_j r_{jif}, f \in T_i) > V_i(\sum_j r^*_{jif}, f \in T_i)$ for all $i \in S$.*

Definition 5 *The feasible allocation* r *has* the core property *with respect to exchange ratios γ if it cannot be blocked by any coalition of peers.*[1]

We first show that the core property is satisfied by any multilateral equilibrium. A multilateral equilibrium allocation cannot be blocked by any coalition of peers, because essentially all peers have already optimized with respect to the same prices.

Proposition 4 *Any multilateral equilibrium allocation has the core property with respect to the equilibrium exchange ratios $\gamma_{ij} = p_i/p_j$.*

We next show that, when a user is interested in one file, a bilateral equilibrium with the core property is a multilateral equilibrium. The insight is similar to Example 3: it can be shown that if no price vector exists such that $\gamma_{ij} = p_i/p_j$, then there must exist users i_1, i_2, ..., i_k such that $\prod_{i=1}^{k} \gamma_{i,(i \bmod k)+1} < 1$. In that case, there is a coalition of k peers that can block the allocation.

Proposition 5 *Suppose $|T_i| = |F_i| = 1$ for all $i \in N$. If a bilateral equilibrium allocation r^* with exchange ratios γ has the core property, then it is also a multilateral equilibrium allocation.*

[1]Note that our definition of the core is distinct from the usual definition in game theory, as it depends on the exchange ratios.

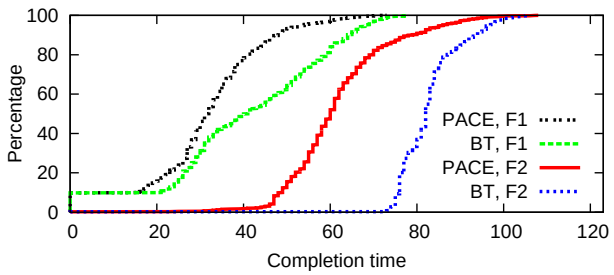

Figure 3: **Completion times for PACE and BitTorrent exchanges. File F1 started at 10% of the nodes; file F2 at a single node.**

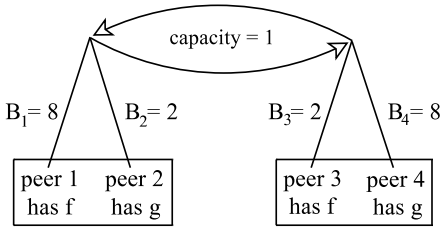

Figure 4: **System with peers $\{1,2,3,4\}$ and files $\{f,g\}$. Peers are located in two clusters; transfers are constrained by bandwidth constraints of peers and the inter-cluster link.**

The results in this section provide a rigorous comparison of bilateral and multilateral exchange in terms of equilibria. Figure 3 provides a comparison of dynamic behavior. The figure shows the completion times of peers under multilateral exchange (PACE) and rate-based tit-for-tat (BitTorrent). We see that peers complete significantly faster in multilateral exchange, demonstrating that it behaves well in dynamic settings. Section §7 describes our simulations in greater depth.

2.3 Comparing Pricing Schemes

This section compares three pricing schemes for multilateral exchange: (1) one price per peer (denoted PP); (2) one price per file (denoted PF); and (3) one price per file per peer (denoted PFP). We compare the schemes through their equilibria. First, we show that all three are equivalent when transfers are only constrained by peer upload capacity. However, we then demonstrate that PF may be strictly worse than PP if the network topology is non-trivial. Finally, we show that PP and PFP yield equivalent equilibria, even when the network topology is non-trivial. Since explicitly pricing every file of every peer is much more complicated than only maintaining a single price per peer, our analysis suggests PP is the most desirable scheme.

Our first goal is to compare the PP and PF schemes. Note that the multilateral exchange in the preceding section used the PP scheme. The PF scheme is similar, but peers optimize with respect to per file prices instead, $(p_f, f \in F)$. That is, the last constraint in the Multilateral Peer Optimization problem becomes $\sum_{j,f} p_f r_{jif} \leq \sum_{j,f} p_f r_{ijf}$.

We first show that if transfers are only constrained by the upload capacity of peers, then the PF and PP schemes are equivalent, in the sense that an equilibrium for one scheme exists if and only if an equilibrium for the other exists, and the rate allocations are the same in both equilibria. Let $(p_f^*, f \in F)$ be an equilibrium price vector for the PF scheme. Setting $p_i^* = \max_{f \in F_i}\{p_f^*\}$, we get an equilibrium for the PP scheme, since the optimization problem of each peer does not change. Conversely, let $(p_i^*, i \in F)$ be an equilibrium price vector for the PP scheme. Setting $p_f^* = \min_{i:f \in F_i}\{p_i^*\}$, we also get an equilibrium for the PF scheme.

Given a non-trivial network topology, however, links other than peer access links may be congested. These links need to be priced as well to ensure efficient network usage. Again abusing notation, we denote the price of link ℓ by p_ℓ. For this example, we assume that we can price every link in the network, an assumption we waive in §4. When peer i is downloading from peer j, i pays j, but also all links that i's traffic traverses. Finally, we assume that whatever is paid to traverse links in the network is rebated equally to all users; our results also hold for other rebating schemes. (We return briefly to these network payments in §6.5.) The following example shows that when the network is non-trivial, equilibria may fail to exist under the PF scheme, even though they exist for the PP scheme.

Example 6 *There are four peers and two files, with file allocation and demand as shown in Figure 4. The network has two clusters, consisting of peers $\{1,2\}$ and $\{3,4\}$, with a bidirectional link ℓ of capacity 1 connecting them. Peers $\{1,3\}$ have file f and want file g; peers $\{2,4\}$ have file g and want file f. The peers' upload capacities are $B_1 = B_4 = 8$ and $B_2 = B_3 = 2$.*

This system has no equilibrium under the PF scheme. If $p_\ell = 0$, peers demand $d_1 = \frac{p_f}{p_g}B_1$, $d_2 = \frac{p_g}{p_f}B_2$, $d_3 = \frac{p_f}{p_g}B_3$ and $d_4 = \frac{p_g}{p_f}B_4$ when optimizing. The market clears only if $d_1 + d_3 = B_2 + B_4$, which implies $p_f/p_g = 1$. But then $d_1 = 8$, which is not feasible, since the maximum total rate at which peer 1 can download is 3. If $p_\ell > 0$ and there is a single price per file, then peers only download locally and, from the market clearing condition, we get $p_f/p_g = 4$ in one cluster and $p_f/p_g = 1/4$ in the other, which is a contradiction.

On the other hand, under the PP scheme, there is an equilibrium with prices $(p_1, p_2, p_3, p_4) = (1,3,3,1)$ and $p_\ell = 2$. The download rates of peers are $d_1 = d_4 = 3$, $d_2 = d_3 = 7$. Peers 1 and 4 download at rate 2 locally and at rate 1 remotely from each other. The revenue collected from the link ℓ is rebated equally to all peers, which allows peer 2 to download more than $(p_g/p_f)B_2$.

The preceding example shows that for general network topologies, the existence of a multilateral equilibrium for the PP scheme does not imply the existence of an equilibrium for the PF scheme. Because of the network topology, a file may be uploaded at different prices at different parts of the network. On the other hand, the existence of a multilateral equilibrium for the PF scheme yields an equilibrium for the

PP scheme. A much stronger result holds if we compare the PP and PFP schemes, where prices p_{if} are set on a per-user-per-file basis:

Proposition 7 *For any network topology, there exists a multilateral equilibrium for the PP scheme if and only if there exists a multilateral equilibrium for the PFP scheme.*

We conclude that one price per peer is sufficient to identify heterogeneity in the system. Intuitively, the upload capacity of a peer's access link is the local resource that becomes congested; hence one price per peer suffices for multilateral equilibrium.

The PP scheme provides many practical benefits as well. First, it greatly reduces the number of prices that need be maintained, compared to PFP pricing. Further, price discovery is simplified, especially for unpopular files for which requests are relatively rare. Finally, PP pricing leads to a natural service discipline for uploading files, well-aligned with a peer's incentives: serve requests sequentially and without preemption. The service discipline for PFP pricing is less clear, however: Serving requests only for the highest-priced file may not fully utilize a peer's available resources, while serving requests sequentially is not profit maximizing.

2.4 Dynamics

The preceding section showed that per-peer pricing and per-file-per-peer pricing are equivalent in a static setting in terms of equilibrium (Proposition 7). However, since it is hard to know a system's equilibrium prices or allocation in advance, we need to consider both how downloaders and uploaders are matched (peer discovery), as well as convergence of prices (price discovery). We also briefly discuss the role of the currency in aiding dynamics.

A significant advantage of *explicit* per-peer prices is that they enable fast peer discovery. This short discovery time significantly improves on that needed by systems with *implicit* prices, *e.g.*, such as BitTorrent's rate-based exchange ratios, which have long discovery times. These implicitly-priced systems effectively need to perform a brute-force search across their peers; this has been found in practice to sometimes take tens of minutes [5], and, at least for high-bandwidth peers, requires an asymptotically-linear search to find similar reciprocation rates.

For price discovery, a simple mechanism is to update the prices of peers and links according to the corresponding *excess demand*. In particular, a price should increase if demand exceeds supply, and decrease if demand trails. But how to define supply and demand for a peer? If the PP scheme is employed, a peer's observed demand is the sum of all received rate requests within a fixed time period, and his supply is equal to the upload capacity of his access link. If the PFP scheme is used, excess demand is more complex: a peer calculates demand for each file separately, while his supply for a file is equal to the access-link capacity only if it is his most expensive file, and zero otherwise (as it is optimal for each peer to only upload his most expensive file). Thus the PP scheme leads to simpler price dynamics.

We conclude by briefly discussing the impact of currency on system dynamics. Allowing users to store and exchange currency over time has significant benefits for a system in a dynamic setting. First, peers can engage in trade *before* equilibrium prices are reached. Second, peers can reach a rate allocation by *trading in a decentralized fashion*, without requiring a central authority to clear the market by matching uploaders and downloaders. However, incorporating currency into a system introduces its own complications, since the user experience could potentially be complicated, and peer exchanges and credit balances need to be secured. We demonstrate how these issues can be handled in §5 and §6, respectively.

3 User Incentives

In this section we argue that the use of prices and currency largely provides the correct incentives to users. We begin by noting that, in particular, it encourages efficient use of resources in a *large* P2P system. If the system is large, users cannot accurately anticipate how their actions affect prices, *i.e.*, users have difficulty in predicting the evolution of demand, supply, and prices.

Users are implicitly incentivized both to contribute a high percentage of their upload capacities and to share high-value content, since high-value files will typically increase a user's price. Moreover, with currency stored over time, users are incentivized to contribute even if they are not currently downloading. We also note that a user is paid for delivered, not advertised, rate. (Buyer payments are equal to price $\cdot$ delivered rate $\cdot$ duration.) Thus, a user does not profit by advertising a higher upload capacity than the one he has.

Network prices align user incentives with efficient network usage. If network links are priced correctly, we expect network prices to reflect congestion. Then users internalize their effect on the network, since they have to pay for all network links they use to download. For instance, among peers with the same price, a user prefers to download from the peer with the smallest total network cost, which we expect to correspond to the least congested route. On the other hand, sellers do not benefit from network-cost-related payments, and so the system does not create a perverse incentive for sellers to prefer remote transfers.

One potential concern in a market-based system is the phenomenon of *market power*: Users, either individually or in small cliques, may try to manipulate prices higher than the laws of supply and demand dictate. This effect is mitigated significantly in a market with many sellers, where market manipulation cannot significantly increase profit [19]. To illustrate this, suppose k peers have a file desired by other users in the system. Let $d(p)$ be the demand for the file at price p. Then the equilibrium price p^* satisfies $d(p^*) = \sum_{j=1}^{k} B_j$. Now suppose user i considers increasing his price

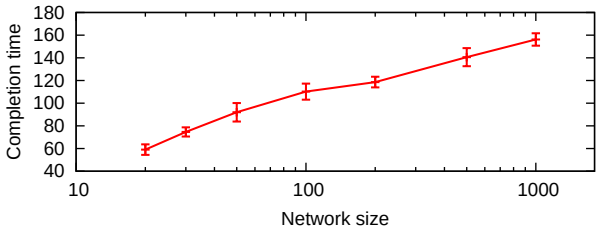

Figure 5: **Completion time grows $O(\log(|\text{network}|))$. Each data point shows the average performance over five runs—error bars show standard deviation—with 10 peers per cluster.**

to $p_i > p^*$ in order to increase his profit. This is beneficial for i only if $p \cdot (B_i - (d(p^*) - d(p))) > p^* \cdot B_i$, or equivalently,

$$\frac{B_i}{d(p)} \geq -\frac{d(p) - d(p^*)}{p - p^*} \cdot \frac{p}{d(p)}$$

Note that $\frac{d(p)-d(p^*)}{p-p^*} < 0$, since as the price increases, the total demand decreases. Assuming $p_i = p^* + \varepsilon$ for some small ε, the right-hand side represents the demand elasticity. Demand elasticity measures price sensitivity and is defined as the percentage change in quantity demanded, divided by the percentage change in price. Thus, user i will not be able to exert market power if demand elasticity $> B_i / \sum_{j=1}^{k} B_j$, *i.e.*, as long as enough sellers compete to upload the file relative to the elasticity of demand.

Market power may still be an issue if only a few users have a file, yet any system can suffer if such users choose to dictate terms to the remainder. In our setting, such users are often the "seeders" of files and *want* to see their content disseminated. Many peers in existing P2P systems exhibit such altruistic behavior.

More generally, sellers *create* other uploaders in the very act of uploading; thus, market power is at best a transient phenomenon, since other sellers quickly emerge as competitors. Further, the number of competitors grows exponentially in time: if a file chunk is always transfered from one user to another in one time period, then after t time periods, $O(2^t)$-times more users will have it. Indeed, this asymptotic behavior is supported by our simulation results (more in §7); Figure 5 shows that the completion time of all nodes to download a file appears to grow logarithmically with network size, as expected.

4 Hierarchical Network Model

Although much of the discussion in §2 assumes that the only resource constraints are the upload capacities of peers, in reality of course other network resources may be congested. As briefly discussed in §2.3, our model can be extended to include one price per link in the network. Downloaders pay the sum of all link prices along a path for content, including the upstream link of the uploading peer; link prices are increased in response to congestion.

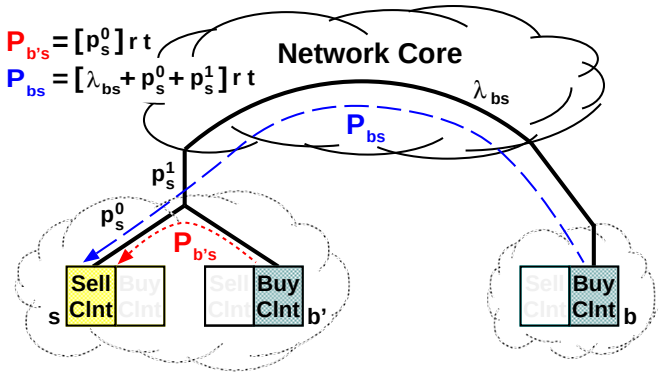

Figure 6: ***Network Model.*** **Clusters of well-connected peers connect across wide-area links. To download file f from sell client s, local buy client b' pays $P_{b's}$ and remote b pays P_{bs} to s.**

However, this approach is probably infeasible in practice, given a lack of network topology and routing information, inaccurate bandwidth capacity estimations, and computational complexity. Thus, we propose a hierarchical network model, outlined in Figure 6, that separately prices most of the bottlenecks that lead to supply constraints.

We observe that, to a first approximation, we can view the entire network as composed of *local clusters*, connected together by the wide-area core. Our model assumes that most bottlenecks—especially those due to transient congestion—are at access links, where we can accurately estimate capacity and adapt prices accordingly. Rare bottlenecks in the core are captured via slowly-changing *network prices*, which act as "shadow prices" for resource constraints in the core. Different network prices may be associated with different pairs of clusters. Section §6 discusses how these network prices can be set, potentially promoting cooperation between ISPs and P2P systems. While less expressive than a price per link, our design compares favorably to typical AS-level clustering approaches [14, 1], which statically restrict peers to their local neighbors. These topological approaches do not capture any dynamic resource constraints at these local peers.

Our system design does not specify precise capacity requirements for local clusters and the wide-area network. Rather, we anticipate that nodes in the same local cluster share relatively high-capacity links to each other (such as within a LAN or a switched university network) and that transmission across the wide area will involve shared, lower-capacity access links to the Internet (*e.g.*, a DSL connection or a university's external link).

To capture these clusters, a peer s therefore maintains two prices, p_s^0 and p_s^1, corresponding to the immediate upstream link at s and the access link shared by all nodes in the same cluster as s, respectively. One could extend this hierarchical model to capture additional choke-points at the network's edge if needed and price these links separately and accordingly. In fact, different peers could maintain different numbers of prices, based on their local network conditions (and this would be largely transparent to buyers). For simplicity,

we restrict our further consideration to these two prices.

If peer b' is within the same local cluster as s, b' pays $P_{b's} = p_s^0 \cdot r \cdot t$ to s for a transfer of duration t at rate r. Such a payment is shown by the red dotted arrow in Figure 6.

If peer b is not within the same local cluster, the traffic must also traverse the access link for s's cluster and the wide-area core. Peer b pays $P_{bs} = (\lambda_{bs} + p_s^0 + p_s^1) \cdot r \cdot t$ to download from s (the blue dashed arrow),[2] where λ_{bs} is the network price between the clusters of b and s. Of this, only $(p_s^0 + p_s^1) \cdot r \cdot t$ is paid to s; the remainder is collected by the system for future rebate (see §6.5).

5 PACE Client Mechanisms

In the preceding sections, we showed how currency-based multilateral exchanges can lead to more efficient systems. One typical complaint against explicit pricing and currency, however, is that the process of setting prices and bidding for goods becomes a usability hurdle. Indeed, this hurdle is seen by the designers of systems such as MojoNation [29] as their major reason for failing to be widely adopted.

We argue that pricing mechanisms, rather than being directly exposed to end-users, can serve as *algorithmic devices* to ensure efficient exchange: We can expose a very simple interface to users—basically, listing their uploadable files and desired downloads—while having users' software optimally compute their buy and sell behavior. However, as we argued earlier in §3, we expect that even strategic users cannot gain any significant benefit from operating in a manner other than that specified by our algorithms.

This section describes the design and algorithmic mechanisms of buy and sell clients in our system, which we call PACE (Price-Assisted Content Exchange). PACE clients interact with each other across both local- and wide-area networks; when chunks of a file are downloaded by a buy client from a sell client, (virtual) currency flows in the opposite direction. We present the accompanying system services and protocols to support their secure interaction next in §6.

5.1 Sell Client

The sell client interface allows the user to declare (1) the files he is willing to upload and (2) the total upload capacity he is willing to commit (either as absolute numbers or as a percentage). The sell client s maintains prices p_s^0 and p_s^1, corresponding to the two edge resources being priced (per §4). This section details the price update rules and service discipline used by the sell client.

The sell client s follows a very simple price update rule in principle. Prices increase if demand exceeds supply and fall if the opposite; we use a multiplicative increase/multiplicative decrease rule, so that convergence is insensitive to the units in which prices are measured. A difficulty arises because there are *two* scarce resources that are being priced, which are *complements* for peers outside the cluster of s.

Sell client s estimates demand and supply over a fixed time interval. To update p_s^1, demand is estimated as the total requested download rate originating at *remote* buy clients, provided they offer at least p_s^1. A request is counted regardless of whether sufficient capacity existed to serve it.[3] Supply is estimated as the sell client's upload capacity on its access link, B_s^1. Estimating B_s^1 may be done by tracking the maximum aggregate throughput ever seen across all uploads. (Note that systems such as BitTorrent similarly use edge-capacity estimation, *e.g.*, to set its active set size [22].)

The approach to update p_s^0 is similar. Demand is estimated by aggregating the rates of all download requests from both local and remote buy clients, taking the access-link bandwidth constraint into consideration. In particular, demand is equal to the sum of local requests and the minimum of remote requests and B_s^1. Supply is the upload bandwidth on the sell client's immediate link, B_s^0.

Finally, we consider the sell client's service discipline. Incoming download requests are served in the order of arrival; they can be immediately notified of acceptance, subject to available capacity constraints ($B_s^{0|1} - u_i^{0|1}$). (Recall that *per-peer* pricing gives the seller no real incentive to queue requests.) This greatly simplifies system design: If we queued requests, then buy clients would need to maintain a large number of outstanding requests, forcing them to reason about spending over a long time horizon.

5.2 Buy Client

The buy client's interface allows the user to choose (1) the files he is interested in (denoted by T_i for user i) and (2) a *savings rate*, *i.e.*, the percentage η of the user's current budget that should be saved for the future. During each fixed time interval, our buy client sets aside a fraction η of the user's bank-account balance,[4] and divides the remainder equally among all files the user wishes to download. For each such file f, the buy client follows a simple algorithm: First, given a set of sell clients that have f, order these clients j by the increasing *total* price the buy client has to pay for downloading from j (including network prices and remote access prices, if applicable). Ties are broken in order of increasing network prices. Then, at each seller j in this order, the buy client spends as much of its budget committed to f as possible. If j's upload capacity is exhausted, the buy client moves to the

[2] Note that any remote b that downloads from s pays to traverse the access link for s's cluster, but not for its own access link. This asymmetry means that congestion on the downstream link from the wide-area core to b's cluster may not be correctly priced. This congestion applies to all remote files, however, and since local prices are typically lower than remote prices, buy clients will generally prefer local sources anyway.

[3] In practice, demand might be somewhat overestimated, as a buy client can issue several sequential requests to different sell clients until a successful download. However, demand is overestimated only when there already is excess system-wide demand, so prices would have increased anyway.

[4] This balance excludes any potential revenues from *anticipated* uploads, since it may not be known what such potential revenues are.

next sell client in the list. The buy client stops when it exhausts its budget for f. Any unused budget is split evenly between files in $T_i - \{f\}$.

This algorithm, though quite simple, can be interpreted through a foundational utility model for the user. In particular, the buy client behaves *as if* a user's utility for downloading is $\sum_{f \in T_i} \log d_f$, where d_f is the total download rate obtained for file f. Let M be the user's current bank account balance and $\bar{p}_f$ be the average price he has to pay for file f over available sellers (in the order described above). Given that the user wants to reserve η percentage of his current budget, the client solves the following maximization problem on the user's behalf:

$$\begin{aligned} \text{maximize} \quad & \sum_{f \in T_i} \log d_f \\ \text{subject to} \quad & \sum_{f \in T_i} \bar{p}_f \cdot d_f \;\leq\; (1-\eta)M \\ & d_f \geq 0, \; \forall f \in T_i \end{aligned}$$

The optimal solution recovers exactly the above algorithm: It is optimal to divide $(1-\eta)M$—the budget allocated to this period—equally among all files in T_i. Note that this does not depend on the relative average prices of files. (Note that although our interpretation is in terms of per file prices, our implementation is in terms of per peer prices.)

There is a tradeoff between simplifying the user's experience and allowing the user to personalize his utility function. The above mechanism favors simplicity over descriptive power; other approaches can prefer the opposite.

6 PACE System Design

This section describes the services and protocols that complete the PACE system. To enable users to act as both buyers and sellers in currency-backed content distribution, PACE should support the following functionality:

1. ***Resource discovery.*** A buy client should learn a set of low cost sell clients from whom to download content.
2. ***Network friendliness.*** Service or network operators can express preferences for peer download behavior across the wide area; buy clients are incentivized to follow these preferences.
3. ***Transaction security.*** Exchanges between two parties should be ε*-fair*: sell clients should receive payment within an ε of the content transmitted for the agreed price; and buy clients should only pay for what they receive.
4. ***Currency security.*** Clients cannot forge system currency nor spend more money than their current balance allows. With online credit checks, attempting such transactions will fail; with offline credit checks, such buy client behavior will be detected promptly.
5. ***User and money management.*** The system concerns itself with user registration and collusion, bootstrapping, and managing the money supply over the long term.

The remainder of this section describes our design for achieving these five properties, organized as above.

System overview. At a high level, PACE is composed of five main components. First, users run both *buy* and *sell clients* to trade content for virtual currency. Sell clients advertise their existence (liveness), shared files, and current price to a *rendezvous service*, where buy clients also connect to discover nearby instances of sell clients offering desired content. A *network price service* specifies the network prices to accommodate network operator preferences, while a *bank* securely tracks each users' accrued capital for a particular currency.

We envision many rendezvous, bank, and network price services to exist simultaneously, run by providers seeking to disseminate *collections* of files. These collections may range from large multimedia libraries (*e.g.*, iTunes and YouTube), to aggregates of user-hosted files (*e.g.*, the PirateBay BitTorrent tracker [21]), to a corpus spanning the entire web (*e.g.*, CoralCDN [8]). We logically separate these services because we envision different ecosystems to arise around all three: rendezvous is file-specific; banks are currency-specific (attached to one or more file collections); and yet network prices are independent of files or currencies.

While some *logical* centralization may be considered undesirable, it is common to many of today's deployed peer-to-peer systems. Virtually all known P2P systems, from Gnutella through Tor, use servers for bootstrapping. Skype uses centralized servers for registration, rendezvous, and accounting. BitTorrent uses a centralized tracker for rendezvous, and its "private trackers" provide (albeit insecure) upload/download ratio accounting for long-term incentives. Even so-called "trackerless" BitTorrent uses a distributed hash table merely to partition responsibility for files across peers, much as one could federate our services. These logically-centralized services can be made scalable: the PirateBay BitTorrent site was, as of January 2008, tracking over one million files and 10 million peers [21]. That said, this section also discusses how some system services may be federated between existing network providers.

Trust requirements. Most of this paper has been focused on deploying peer-assisted content distribution in strategic or mutually-distrustful settings *i.e.*, where peers may run their own buy and sell client implementations. The additional system components we now introduce require stronger, although still common, trust assumptions. We assume some trusted one-of-band mechanism by which clients learn a file's meta-data for later chunk-based integrity verification—much like BitTorrent's need for *.torrent* files. We trust the rendezvous service to properly return sell client identities, which otherwise would affect content availability—as would faulty BitTorrent trackers. Finally, we assume that the bank

performs currency accounting—much as private trackers are assumed to do ratio accounting.

There may exist compelling settings where *end-clients may be trusted* to obey the protocol, however, such as with set-top boxes. In these cases, in being able to assume client altruism, the system can avoid the use of currency and banks; even then, however, our explicit prices would serve as a mechanism for efficient resource discovery and allocation. While this paper does not consider this setting further, it may become the dominant deployment platform for video-on-demand. These dedicated hardware/software platforms are becoming increasingly popular—*e.g.*, Comcast On-Demand, Verizon FiOS TV, Microsoft Xbox Live, AppleTV, the Netflix/LG partnership, and so on—with peer-assisted delivery on the horizon.

6.1 Resource and Price Discovery

We do not fully prescribe how buy clients efficiently discover sell clients. In practice, there are a number of feasible engineering designs that PACE implementations could use, from logically-centralized trackers (a la BitTorrent), to a federated application-level anycast service (*e.g.*, OASIS [9]), to distributed hash tables indexing clients' addresses [8]. Because each user only needs to publish a single price (or one per hierarchical access link, given our network model), prices can be stored within the rendezvous service without significant communication overhead for updating. Thus, a query to the rendezvous service—which takes a file identifier and buy client information as input—can immediately return a randomized set of IP addresses with minimal published cost, potentially biased by network cluster locality or inflated by additional network prices, which we discuss next.

Upon discovering a set of sell clients with chunks of interest, the buy client determines their latest prices, allocates its budget across desired files, and prioritizes transfers which cost less, per §5.

6.2 Network Friendliness

Beyond leveraging users' prices to minimize resource congestion, PACE benefits network operators by having clients avoid expensive network links whenever possible. For a transfer traversing the wide-area Internet, the *network price* λ is added to the sell client's file price p_s, per §4. Thus, to calculate the total price for downloading from a specific sell client, a buy client needs to discover the network price between the clients' clusters.

PACE implementations and deployments have various design choices here as well. Well-suited for immediate deployment, a third-party service could independently determine network prices via network measurements [9, 18] or explicit ISP input. Alternatively, a buy client's network provider could run a lookup service itself (as with DNS and DHCP) and use these network prices to perform *policy-based traffic engineering* on its intra-ISP and egress traffic, taking its AS peering and transit relationships into account. In the longer term, we could imagine such network prices may be propagated *between* ASes, perhaps alongside BGP announcements, and thus enable ISPs to better express their preferences for P2P data transfers. This inter-ISP final proposal introduces interesting incentive questions when coupled with current billing practices—*e.g.*, customer ASes trying to offload traffic and providers attempting to attract traffic to their customers—so we leave this to future work.

Network prices should not be too high, however. Otherwise, a large fraction of a transfer's costs would go to the network, which would need special care by some subsequent money management task (per §6.5). Thus, a stand-alone service should choose network prices from a limited range.

In the preceding section, we proposed that the rendezvous service could take sell clients' prices into account when determining which clients to return. There are various design choices to be made, however, when incorporating network prices into this model. For a loose coupling between the two services, the rendezvous service may only incorporate peers' file prices or coarse-grained clustering information in its selection—the latter akin to coarse-grained locality proposals for peer-assisted CDNs [14, 1]—after which buy clients would subsequently lookup network prices themselves. This is well-suited for federated deployments in which ISPs provide network price services. On the other hand, if rendezvous and network price services are more tightly coupled, the exact network price could be considered in the selection criteria, leading to greater efficiency.

6.3 Securing Transactions

Exchanges between buy and sell clients should be ε-*fair* in that the goods they ultimately exchange are equal (within some small ε factor) with respect to the agreed-upon payment P. That is, if one party fails during the transaction after only $\frac{i}{k}$th of the content is transferred, and $\varepsilon = 1/k$, then the buy client should pay $\frac{i-1}{k}P \leq P' \leq \frac{i}{k}P$.

We achieve this property through the following. At the start of an exchange for a chunk, the buy client b cryptographically commits to a payment, which includes the final output of a cryptographic hash chain of length k. Then, as the sell client s begins uploading content to b, b responds with a stream of micropayments (akin to [24]), each a successive pre-image of the hash chain. The buy client only sends a micropayment after receiving another $1/k$-th of the chunk,[5] while s stops transmitting data soon after b stops responding with micropayments. The buy client can thus "steal" bandwidth from at most a small transmission window before it is detected (and blacklisted locally by s).

The ε-fair protocol. More formally, to initiate a chunk

[5] If file meta-data only specifies the hashes of chunks, then a *malicious* (as opposed to a strategic) s could send a bad chunk, but b could only detect this—and subsequently blacklist s—after downloading the chunk. If this is a concern, to strictly achieve the ε-fair definition, s could send hashes corresponding to $(1/k)$-range subchunks at the transfer's inception, which are incrementally verified by b before he responds with micropayments.

transfer, sell client s sends a counter c_t and the agreed-upon price p to buy client b, who generates a commitment to s:

$$\sigma \leftarrow \mathsf{sign}_b(b,\ s,\ P,\ \Lambda,\ k,\ h_0,\ \hat{h},\ c_t)$$

where sign is a cryptographic signature with message recovery in the "hash-then-sign" paradigm [4]. $P = prt = p \cdot |chunk|$ is the full payment for the chunk, of which Λ is the network cost. This payment P can be split across k micropayments. h_0 and $\hat{h}$ are outputs of a cryptographic hash function, generated as follows. The buy client first selects some random string h_k, then recursively computes a hash chain: $h_{i-1} \leftarrow \mathsf{hash}(h_i)$ for $i = k \ldots 1$. The buy client also creates a "shortcut" $\hat{h} \leftarrow \mathsf{hash}(\mathsf{S}, h_k)$ for global constant S.

At the beginning of a transfer, the buy client sends σ and its certified public key (per §6.4) to s. s verifies both b's public key and σ's contents and signature. The sell client continues to transmit only if it receives a successive pre-image of the hash chain $h_i = \mathsf{hash}^{-1}(h_{i-1})$ every $\frac{1}{k}$-th of the transfer.

To deposit a payment at a bank (discussed next), b provides the tuple $\langle h'_i, i, \sigma \rangle$. The bank verifies σ and that the statement is not being replayed (using the counter c_t, also discussed next). If these checks succeed, the bank determines whether $i = k$, in which case the bank checks $\hat{h} \stackrel{?}{=} \mathsf{hash}(\mathsf{S}, h'_i)$. If $0 < i < k$, the bank verifies $h_0 \stackrel{?}{=} \mathsf{hash}^i(h'_i)$, *i.e.*, recursively applies the hash function i times. If this check passes, the bank executes the transaction for an amount $P' = \frac{i}{k}P$; that is, it debits b with the amount P', credits s with $P' - \frac{i}{k}\Lambda$, and reserves the (potentially zero) payment $\frac{i}{k}\Lambda$ for network costs.

Thus, we see that the bank's computational overhead is one "double-entry book-keeping" transaction, one signature verification, and one cryptographic hash in the normal case. In the case of truncated transfers, this overhead increases to at most k hashes, instead of one. (This use of $\hat{h}$ is a pure computational optimization.)

Aggregating buyer payments. To further reduce the load at a bank, sell clients can aggregate multiple payments from the same buy client before depositing them as one transaction. Repetitive behavior is especially likely once two well-located peers discover one another and transfer many chunks, *e.g.*, those belonging to the same network cluster.

Specifically, upon initiating a subsequent transfer, s can send any yet-to-be-deposited σ's to the buy client, to have information from these messages $\langle P,\ \Lambda,\ \hat{h},\ c_t \rangle$ included in the latest commitment. Note that the inclusion of the counter c_t for replay protection protects the buy client from double counting. In this case, when depositing ℓ transfers, the bank still performs a single transaction, one signature verification, and ℓ hashes. If b omits these previous payments in its latest commitment, s can deposit them separately.

6.4 Securing Currency and Balances

To broker the exchange of files belonging to some *collection*, a content provider both runs a bank which manages currency used by the collection, and acts as a user registration authority for peers seeking to download files from the collection. The set of files that belong to a collection can be dynamic; banking/registration providers could thus offer their services to multiple publishers of collections.

Users are identified by public/private key pairs. When registering a new user, the bank issues a signed certificate to the user (with a short expiry time) attesting to its membership. This certificate is used by sell clients to verify the membership of buy clients without online checks. Banks must therefore refresh the certificate of active clients every expiry period; these updates can be piggy-backed on other control traffic, *e.g.*, deposits.

For each registered user, the bank stores, at a minimum, a hash of its public key, its current balance, and a monotonically-increasing counter c_{last} of its last deposit. To prevent sell clients from replaying deposits, the counter c_t included in a deposited payment must be strictly greater than the buy client's c_{last}. If the deposit succeeds, c_{last} is set to c_t.

On the other hand, the bank must also prevent buy clients from issuing payments for money they do not have. For increased scalability, we have suggested a model by which individual sell clients can aggregate payments over short time horizons into single deposits. Of course, there then exists some window of vulnerability during which buy clients may overdraw on their accounts. Thus, if the bank ever detects that a client maintains a negative balance in its account, it should suitably punish or evict that client—by not renewing its membership certificate—from its network.

One could, of course, reduce this potential fraud by minimizing both sell clients' aggregation of payments and buy clients' certificate expiry times. In the limit, sell clients could perform an *online* check at the start of a transfer (escrowing P for some duration), and avoid any potential for overspending, at the cost of greater bank load and client latency.

We believe that these logically-centralized banks can be easily scaled via traditional replication and partitioning strategies; indeed, the systems community has several decades of experience building highly-scalable and available transaction-processing systems. Further, given that currency is only virtual, one may be able to slightly weaken consistency or freshness guarantees. Finally, as PACE's network model specifically encodes the hierarchical network structure for ISP-friendliness, network providers may deploy (currency-agnostic) local banks to aggregate local transactions to reduce load, especially given the extent to which transactions in PACE are executed locally. This layer of federation adds additional complexity, so we leave its full consideration to future work.

6.5 Managing Users and Money

We conclude by discussing several issues with managing user identities and money as a long-term stored value.

Until now, we have assumed that clients do not collude with one another, either as cliques of multiple real-world

users or as part of a Sybil attack [7]. While we might cite the traditional techniques to limit the scale of such attacks—such as analyzing "introduction" networks (*e.g.*, SybilGuard [31]), leveraging real-world scarce resources (*e.g.*, phone numbers as in Google's GMail), or even linking membership to real-world money (*e.g.*, for subscription-based services)—the limited benefit of collusion in our setting bears some additional comments.

In PACE, a clique of clients cannot increase its aggregate wealth by combining trade amongst themselves with service to non-colluding clients at market rates; this is a consequence of the core property discussed in §2.2, together with Proposition 4. This differs greatly from many reputation systems or systems based on download/upload rates, where shilling reputation announcements or faking content transfers [16], respectively, can increase a clique's aggregate balance.

We now briefly discuss the management of PACE's money supply. As users join and leave, and the network size changes, a bank must ensure that an appropriate amount of currency remains in the system. While the MIMD adaptation of prices makes the system somewhat robust to moderate inflation and deflation, excessive inflation may cause price convergence to take too long, while excessive deflation can lead to insufficient liquidity. However, our logically-centralized bank is in a prime position to control the money supply. We are already assuming that the banks rebates network cost payments to peers. We currently suggest a proportional "tax" on client's balances to prevent hoarding, and reinjecting money into the system via equal rebates to all parties. We leave a careful analysis of money management and rebating schemes to future work.

Finally, we note that the system must have a method by which to bootstrap new users. If Sybil attacks are less of a concern, new users can be granted money upon joining the system as a bootstrapping method. Alternatively, a new user can be offered the ability to download content for free from the bank's operator: The redistribution of this in-demand content—not necessarily desired by the new user itself—can enable the system to provide better quality-of-service for its published content, while allowing users to earn initial capital by contributing upstream resources.

We conclude by noting that while these "free money" approaches may be susceptible to strategic Sybils—unless Sybil attacks are adequately protected by the registration process—-these same problems exist when bootstrapping most other incentive schemes of which we are aware, even if these systems do not explicitly use currency. Indeed, BitTorrent's performance reliance on seeders and its price-discovery mechanism (optimistic unchoking) are precisely "free money" and have been attacked as such [17, 26]. But because PACE maintains currency as stored value, this bootstrapping phase only occurs when *new users* join the system, not for *every file* being distributed and *every bilateral exchange* being transacted.

7 Simulation Analysis

In this section, we wish to evaluate the following hypotheses through simulation. (1) File prices reflect resource constraints. (2) Contributing upstream capacity improves a user's performance (and thus incentivizes such behavior). (3) File prices yield efficient dissemination across multiple files. (4) Buy clients prefer more efficient network links.

Simulator design and configuration. We model network connectivity and capacities using a hierarchical topology generated by BRITE [20], with clusters of nodes connected by AS-level links. In the following experiments, capacity between local hosts is all 100 units/round, while that across wide-area links follows a heavy-tailed distribution on $(10, 1024)$. Each file is comprised of 50 chunks, each of fixed size (25 units). We use these generated graphs to compute end-to-end network capacities; for simplicity, however, we model all pairs of nodes as having independent links, *i.e.*, we do not model cross-traffic congestion. We also assume simple fixed transmission rates (*e.g.*, no TCP slow start). Network prices are static and computed as an inverse logarithmic function of capacity, ranging $[0, 5)$. Money collected from network prices are rebated equally to all peers.

The 6,000-line Python simulator operates in synchronous time steps, with PACE buy and sell clients behaving as specified in §5. Our BitTorrent-like implementation, used for some comparisons with PACE (*e.g.*, in Fig.3), captures the main aspects of the BitTorrent protocol: optimistic unchoking (with 2 slots), active set sizing (with $max(4, \sim\sqrt{B^{\downarrow}})$ slots), and rate-based tit-for-tat with equal-split bandwidth allocation (all per [22]). Both protocols use a local-rarest-first policy for chunk selection.

We first evaluate how the system handles a flash-crowd for a single file: All users simultaneously become interested in the same file, initially published at a single, random sell client (with an initial price of 1). The following results are for a network of 500 peers, comprised of 50 clusters of 10 nodes each. All nodes begin with 1,000 currency units. A randomly-chosen 50% of nodes are "freeloaders," *i.e.*, they never upload content. Non-freeloaders upload—and thus accrue capital—even after finishing to download their file(s). We later consider the multiple file case.

The system behaved similarly when nodes' initial currency varied from 100 to 10,000 units: the equilibrium price just shifts accordingly. However, too little initial currency (10 units) led to liquidity problems, in that many exchanges executed at a 0 price, which led to indistinguishable performance between freeloaders and non-freeloaders; too much currency (1M units) took too long for price convergence, leading to the same result.

Experimental results. We now seek to demonstrate that our system fulfills the above four hypotheses.

Figure 7 plots the prices of sell clients' local (top) and remote (bottom) links. Given their smaller capacity, remote link prices remain higher for longer. Once chunks are dis-

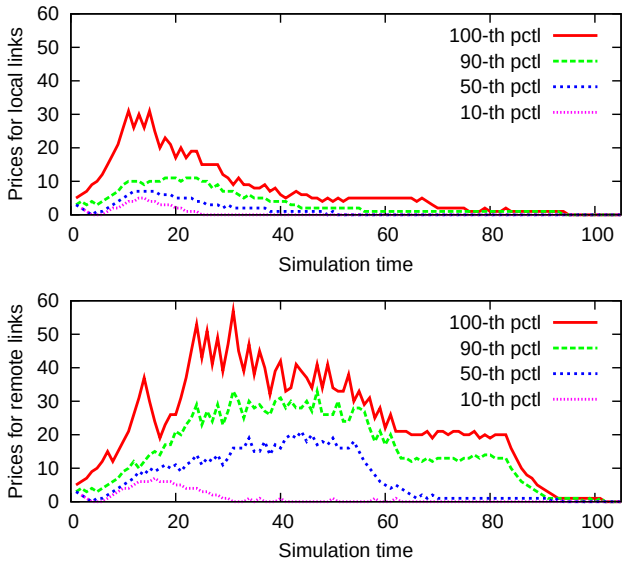

Figure 7: **Local links (top) are cheaper than remote (bottom).**

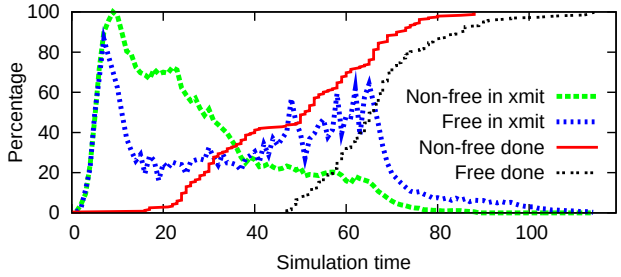

Figure 8: **Non-freeloaders (dashed green on left) download chunks earlier than freeloaders (dotted blue), leading to earlier completion times for non-freeloaders (solid red) than freeloaders (dotted black).**

seminated to a few peers per cluster, local supply can largely satisfy local demand, and local prices drop. We can observe convergence around the remote equilibrium price between time 25 and 85 (the jagged edge corresponds to the MIMD oscillations around the equilibrium price).

Figure 8 gives these prices' performance implications. There are three distinct regions: In the first $\sim$10 time slots, the rate of chunk transmissions for both non-freeloaders and freeloaders greatly increases, as prices are still low (see Figure 7) and every node still has starting capital. Between time 10–40, non-freeloaders download significantly more chunks than freeloaders: As prices are non-zero, only non-freeloaders accrue capital to afford such. Finally, non-freeloaders begin to finish, and prices drop towards zero given the enlarged supply; freeloaders take advantage of this excess capacity (as is socially efficient) and their transmissions increase again. The result: non-freeloaders complete in about 70% of the time.

Figure 9 demonstrates the system dynamics for multiple files. Here, we initiate our 500-node network (no freeloaders) with two files, F1 and F2. Given increased supply of

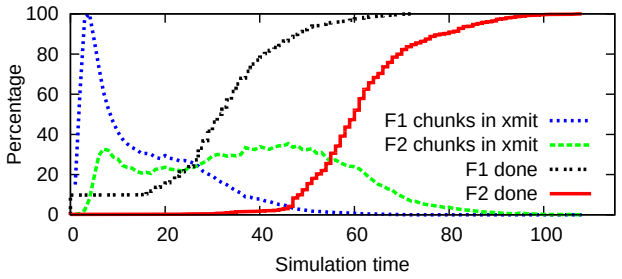

Figure 9: **Transfer rates for two files in the same network. File 1 starts at 50 nodes; file 2 at 1 node.**

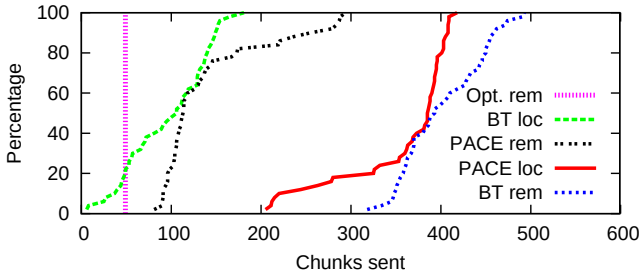

Figure 10: **PACE downloaders prefer local (*loc*) to remote (*rem*) transmissions, but BitTorrent peers (*BT*) makes poor use of locality.**

F1 exists—it starts on 10% of nodes—its transmission rate initially shoots up. While the nodes initiated with F1 start downloading F2 immediately (F2's initial bump in "chunks in transit"), most nodes delay while they download F1: F1's supply and lower available prices have led to saturated downstream links. As individual nodes finish downloading F1, they begin to download F2. Resource allocation thus properly adapts to constraints, yielding a median completion time for F1 that is 50% of F2.

Finally, Figure 10 shows that buy clients prefer to download locally. We return to the single-file case, again with no freeloaders. The graph plots a CDF of chunk transmission frequency (*i.e.*, how many times each chunk traverses local and remote links). The minimum number that each chunk needs to be remotely transfered is 49, which would lead to 450 local transfers (given 50 clusters, 500 users, and 1 initial publisher). Our graph shows a median number of remote transfers 131% greater, primarily caused when chunks sent at low rates (and hence multiple rounds) are concurrently downloaded by multiple peers in the same cluster. Conversely, median local transfers occur at 86% of optimal.

Our BitTorrent simulation, however, performed significantly worse. Figure 10 shows BitTorrent's remote transmissions were 702% greater than optimal, or 26% of the optimal for local. One could add some static locality to BitTorrent [5, 1], but these do not capture dynamic constraints. Adapting to these dynamic constraints is useful; recall that Figure 3 shows PACE outperforming BitTorrent when it had to allocate resources across files. We omit further comparisons between PACE and BitTorrent due to space limitations.

8 Related Work

Early P2P systems did not provide any incentives for participation, leading to extensive freeloading. According to [12], 85% of Gnutella users were sharing no files. The P2P community responded with mechanisms to prevent freeloading by incentivizing sharing.

One approach is to design a system based on *bilateral barter*, as used by BitTorrent [6] and its variants [22, 30], where users can achieve better download performance from peers to which they are simultaneously uploading. Not only are there no network considerations, but users are also not incentivized to continue uploading a file after they finish downloading it, making such systems ill-suited for anything but flash crowds for very large files. Our system encourages uploading, as this builds a user's budget for future downloads. Finally, [23] proposes a volume-based tit-for-tat that is coordinated through a relatively small set of intermediaries, but these peers were not observed to generate adequate demand to create sufficient liquidity.

Another option is monetary incentives [11, 28]: A user's budget decreases when downloading a file and increases when uploading. MojoNation allowed users to price individual transactions in a centralized auction, but the usability hurdle for doing so—which is instead hidden from users in our design—was seen as its downfall [29]. Dandelion [27] describes currency-backed exchanges that use an online centralized bank, but gives no consideration about the resulting market, *i.e.*, how prices are set or adapt. Kash *et al.* studies performance as a function of the total amount of internal currency available [15], but does not consider heterogeneity in the system. While the market-theoretic formulation of [2] considers files with different prices, it does not propose a system design. Further, none of these approaches consider efficient resource utilization; in particular, no pricing is used for communication constraints between peers.

9 Conclusions

This paper studies the role of prices in peer-assisted content distribution. Our novel theoretical results provide insight into the gap between bilateral and multilateral exchange, and demonstrates how relatively simple pricing mechanisms are sufficient for efficient allocations. Given these results, we present PACE, a system for currency-backed content exchange. Beyond efficient use of network resources, PACE's algorithmic mechanisms are promising to hide complexity from users, provide robustness to strategic deviations, incorporate network friendliness, and prevent cheating.

Beyond a more in-depth consideration of ISP-propagated network prices and long-term money management, one other aspect for future work is interesting: the role prices could play in server provisioning. Content providers seek to use peer-assisted content distribution to cut costs, yet they also seek a certain quality-of-service. In PACE, providers can use prices, much like users do, to allocate their server resources near optimally. An interesting question remains how providers can use these prices to determine the level of provisioning sufficient to achieve desired QoS guarantees.

Acknowledgments. We thank Matthew Ceasar, Rodrigo Fonseca, Ashish Goel, David Mazières, Jennifer Rexford, and Benjamin Van Roy for helpful discussions and comments on earlier drafts of this paper.

References

[1] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPs and P2P users cooperate for improved performance? *ACM CCR*, 37(3), 2007.

[2] C. Aperjis and R. Johari. A peer-to-peer system as an exchange economy. In *GameNets*, 2006.

[3] E. Bangeman. P2P responsible for as much as 90 percent of all 'Net traffic. *ArsTechnica*, Sep 3 2007.

[4] M. Bellare and P. Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In *EUROCRYPT*, 1996.

[5] R. Bindal and P. Cao. Can self-organizing P2P file distribution provide QoS guarantees? *OSR, Self-Organizing Systems*, 2006.

[6] B. Cohen. Incentives build robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[7] J. R. Douceur. The sybil attack. In *IPTPS*, 2002.

[8] M. J. Freedman, E. Freudenthal, and D. Mazières. Democratizing content publication with Coral. In *NSDI*, 2004.

[9] M. J. Freedman, K. Lakshminarayanan, and D. Mazières. OASIS: Anycast for any service. In *NSDI*, May 2006.

[10] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, analysis, and modeling of BitTorrent-like systems. In *IMC*, 2005.

[11] M. Gupta, P. Judge, and M. Ammar. A reputation system for peer-to-peer networks. In *NOSSDAV*, 2003.

[12] D. Hughes, G. Coulson, and J. Walkerdine. Free riding on Gnutella revisited: The bell tolls? *IEEE Dist. Systems Online*, 6(6), 2005.

[13] S. Jun and M. Ahamad. Incentives in BitTorrent induce free riding. In *WEIS*, 2005.

[14] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should Internet service providers fear peer-assisted content distribution? In *IMC*, 2005.

[15] I. Kash, E. J. Friedman, and J. Y. Halpern. Optimizing scrip systems: Efficiency, crashes, hoarders, and altruists. In *EC*, 2007.

[16] Q. Lian, Z. Zhang, M. Yang, B. Zhao, Y. Dai, and X. Li. An empirical study of collusion behavior in the Maze P2P file-sharing system. In *ICDCS*, 2007.

[17] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer. Free riding in BitTorrent is cheap. In *HotNets*, 2006.

[18] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An information plane for distributed services. In *OSDI*, 2006.

[19] A. Mascolell, M. D. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[20] A. Medina, A. Lakhina, I. Matta, and J. Byers. Boston University Representative Internet Topology Generator, 2007.

[21] T. Mennecke. The Pirate Bay breaks 10 million users. *Slyck News*, Jan 26 2008.

[22] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Do incentives build robustness in BitTorrent? In *NSDI*, 2007.

[23] M. Piatek, T. Isdal, A. Krishnamurthy, and T. Anderson. One hop reputations for peer-to-peer file sharing workloads. In *NSDI*, 2008.

[24] R. L. Rivest and A. Shamir. PayWord and MicroMint–Two simple micropayment schemes. In *Workshop on Security Protocols*, 1997.

[25] S. M. Ross. *Introduction to Probability Models*. Academic Press, 2007.

[26] M. Sirivianos, J. H. Park, R. Chen, and X. Yang. Free-riding in BitTorrent networks with the large view exploit. In *IPTPS*, 2007.

[27] M. Sirivianos, J. H. Park, X. Yang, and S. Jarecki. Dandelion: Cooperative content distribution with robust incentives. In *USENIX Technical*, 2007.

[28] V. Vishnumurthy, S. Chandrakumar, and E. G. Sirer. KARMA: A secure economic framework for P2P resource sharing. In *WEIS*, 2003.

[29] B. Wilcox-O'Hearn. Personal Communication, 2007.

[30] F. Wu and L. Zhang. Proportional response dynamics leads to market equilibrium. In *STOC*, 2007.

[31] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. SybilGuard: Defending against Sybil attacks via social networks. In *SIGCOMM*, 2006.

A Proofs of Pricing Results

Proof of Proposition 2: Substituting $\gamma_{ij} = p_i/p_j$ in the Bilateral Peer Optimization problem, we get the constraints

$$p_j \sum_f r_{jif} = p_i \sum_f r_{ijf} \text{ for all } j.$$

We observe that if r is feasible for the Multilateral Optimization problem of peer i, then we can construct $\bar{\mathsf{r}}$ such that (i) $\bar{\mathsf{r}}$ is feasible for the Bilateral Optimization problem of peer i, and (ii) $\bar{\mathsf{r}}$ gives the same utility as r to user i. In particular, we can achieve this by setting $\bar{r}_{jif} = r_{jif}$ for all j, f and then choosing $\bar{r}_{ijf}$ for each j such that $\sum_f \bar{r}_{ijf} = (p_j/p_i)\sum_f \bar{r}_{jif}$.

The previous observation holds for any feasible solution of the Multilateral Optimization problem, and thus it also holds for an optimal solution. This implies that an optimal solution for the Bilateral Peer Optimization problem is also optimal for the Multilateral Peer Optimization problem. We conclude that a bilateral equilibrium allocation with exchange ratios γ_{ij} that satisfy $\gamma_{ij} = p_i/p_j$ for all $i, j \in N$ is also a multilateral equilibrium allocation. ■

Proof of Proposition 4: Consider a competitive equilibrium and suppose that there exists a coalition S that blocks it and let r be the corresponding rate allocation. Then by Definition 4 and the Multilateral Optimization problem,

$$\sum_{j,f} p_j \cdot r_{jif} > p_i \cdot B_i, \forall i \in S.$$

Summing over all $i \in S$,

$$\sum_{i\in S}\sum_{j\in S} p_j \cdot \sum_f r_{jif} + \sum_{i\in S}\sum_{j\notin S} p_j \cdot \sum_f r_{jif} > \sum_{i\in S} p_i \cdot B_i. \quad (1)$$

Since $\gamma_{ij} = p_i/p_j$, condition (iii) of Definition 3 becomes $\sum_{j\notin S} p_j \cdot \sum_f r_{jif} = p_i \cdot \sum_{j\notin S}\sum_f r_{ijf}$ for all $i \in S$, $j \notin S$. Substituting in (1)

$$\sum_{i\in S}\sum_{j\in S} p_j \cdot \sum_f r_{jif} + \sum_{i\in S}\sum_{j\notin S} p_i \cdot \sum_f r_{ijf} > \sum_{i\in S} p_i \cdot B_i.$$

By condition (ii) of Definition 3, $\sum_{f,j\in S} r_{ijf} \le B_i - \sum_{f,j\notin S} r_{ijf}$, thus

$$\sum_{i\in S} p_i \cdot \sum_{f,j\in S} r_{ijf} > \sum_{i\in S} p_i \cdot \sum_{f,j\in S} r_{ijf},$$

which is a contradiction. ■

Proof of Proposition 5: We first show that in a bilateral equilibrium, the exchange ratios only depend on the files being exchanged, not on the peers' identities. Suppose that $f \in F_{i_1}$, $f \in F_{i_2}$, $g \in F_{j_1}$ and $g \in F_{j_2}$ and at the bilateral equilibrium, peers i_1 and j_1 exchange f and g, and i_2 and j_2 exchange f and g. At an equilibrium, each peer chooses to exchange only with peers to whom he has the best exchange ratio. This yields the following inequalities:

$$\gamma_{i_1,j_1} \ge \gamma_{i_1,j_2} \; ; \; \gamma_{j_1,i_1} \ge \gamma_{j_1,i_2} \; ; \; \gamma_{i_2,j_2} \ge \gamma_{i_2,j_1} \; ; \; \gamma_{j_2,i_2} \ge \gamma_{j_2,i_1},$$

since i_1 exchanges with j_1 at equilibrium, not j_2; j_1 exchanges with i_1, not i_2; etc. Combining these inequalities with the fact $\gamma_{ij} = 1/\gamma_{ji}$,

$$\gamma_{i_1,j_1} = \gamma_{i_1,j_2} = \gamma_{i_2,j_1} = \gamma_{i_2,j_2}.$$

In a slight abuse of notation, in a bilateral equilibrium we define γ_{fg} to be the unique value of the exchange ratio between any two peers i and j, such that i uploads file f to j and downloads file g from j; again, we obviously have $\gamma_{fg} = 1/\gamma_{gf}$.

We consider a graph with a node for every file and draw an edge between two files if there are two peers exchanging them in the bilateral equilibrium. If the product of exchange ratios on every cycle of this graph is equal to one, then we get a unique price for each file in the following way. We start from a file (say f_1) whose price we set equal to 1 and then take a minimum spanning tree of the graph. We move along the edges of this tree and set prices for other files, so that the exchange ratios are satisfied. In this way we get prices p_f for each file f. Then we can derive prices per peer by setting $p_i = p_f$ for $f \in F_i$. These prices are uniquely defined, since

$|F_i| = 1$ for all i, and satisfy $\gamma_{ij} = p_i/p_j$ whenever i and j exchange at the bilateral equilibrium.[6] Since $|F_i| = |T_i| = 1$ for all peers i, the optimization problem of peer i is not affected by exchange ratios γ_{ij} to peers j with $F_j \neq T_i$. Hence, there exist prices p_i for all $i \in N$ and exchange ratios γ'_{ij} for $i, j \in N$ such that the bilateral optimization problem for each peer is the same under γ and γ', and $\gamma'_{ij} = p_i/p_j$ for all $i, j \in N$. Proposition 2 implies that the bilateral equilibrium allocation is also a multilateral equilibrium allocation.

Now suppose there a cycle of files $f_1, f_2, ..., f_k$ such that $\prod_{i=1}^{k} \gamma_{f_i, f_{(i+1) \bmod k}} < 1$. We will show that in this case there exists a coalition of peers that blocks the bilateral equilibrium allocation. For the remainder of this proof we will denote by f_{i+1} and f_{i-1} the files after and before file f_i *with respect to the cycle*. Necessarily there is an exchange between files f_i and f_{i+1} for $i = 1, ..., k$. We choose a coalition of peers $S = \{1, 2, ..., k\}$ such that peer i uploads file f_i and downloads file f_{i+1}. Note that peers in S are not exchanging with each other at the bilateral equilibrium, because $|T_i| = 1$ for all i.

We will demonstrate that all peers in S can strictly increase their download rates by reducing their upload rates to peers outside S and allocating the remaining rates among themselves. Because of the preferences, peer i will upload to peer $(i-1)$. By sending $r_{i,i-1,f_i}$ to peer $i-1$, peer i reduces the rate he gets from outside S by $\gamma_{f_i,f_{i+1}} \cdot r_{i,i-1,f_i}$. So, the coalition increases i's utility if and only if the rate he receives from S, *i.e.*, $r_{i+1,i,f_{i+1}}$, is greater than $\gamma_{f_i,f_{i+1}} \cdot r_{i,i-1,f_i}$. To show that S blocks r^*, it suffices to find

$$r_{i+1,i,f_{i+1}} \leq B_{i+1}, \forall i \in S \tag{2}$$

such that

$$r_{i+1,i,f_{i+1}} > \gamma_{f_i,f_{i+1}} \cdot r_{i,i-1,f_i}, \forall i \in S. \tag{3}$$

For $\delta, \varepsilon > 0$, we set the following:

$$\begin{aligned} r_{1,k,f_1} &= \delta \\ r_{i+1,i,f_{i+1}} &= \gamma_{f_i,f_{i+1}} \cdot r_{i,i-1,f_i} + \varepsilon, \quad \forall i \in S \end{aligned}$$

Since $\prod_i \gamma_{f_i,f_{i+1}} < 1$, it is possible to choose δ and ε small enough so that conditions (2) and (3) are satisfied. We conclude that if the product of exchange ratios along some cycles is not equal to one, then the allocation is not in the core. ■

Proof of Proposition 7: First suppose that there exists a multilateral equilibrium for the PP scheme with equilibrium prices $(p_i^*, i \in N)$ for peers and the (p_ℓ^*) for links in the network. Then, the peer price vector $(p_{if}^*, i \in N, f \in F_i)$ with $p_{if}^* = p_i^* \ \forall f \in F_i$ and the link price vector (p_ℓ^*) constitute a multilateral equilibrium price vector for the PFP scheme. This holds because it gives rise to the same demand and supply as $(p_i^*, i \in N)$ and (p_ℓ^*).

[6] This is closely related to the following result on Markov chain reversibility: A Markov Chain is reversible if and only if the transition probabilities of forward and reversed chains are equal [25].

Now suppose that there exists a multilateral equilibrium for the PFP scheme with prices $(p_{if}^*, i \in N, f \in F_i)$ and (p_ℓ^*). For each peer i, set $p_i^* = \max_{f \in F_i}$. Peer i only "supplies" his most expensive files at equilibrium. Hence, the prices (p_i^*) and (p_ℓ^*) yield the same demand and supply as (p_{if}^*) and (p_ℓ^*) for each peer. ■