Auction, but Don't Block

Xiaowei Yang
Department of Computer Science
University of California, Irvine
xwy@uci.edu

ABSTRACT

This paper argues that ISP's recent actions to block certain applications (e.g. BitTorrent) and attempts to differentiate traffic could be a signal of bandwidth scarcity. Bandwidth-intensive applications such as VoD could have driven the traffic demand to the capacity limit of their networks. This paper proposes to let ISPs auction their bandwidth, instead of blocking or degrading applications. A user places a bid in a packet header based on how much he values the communication. When congestion occurs, ISPs allocate bandwidth to those users that value their packets the most, and charge them the Vickrey auction price. We outline a design that addresses the technical challenges to support this auction and analyze its feasibility. Our analysis suggests that the design have reasonable overhead and could be feasible with modern hardware.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.6 [**Computer-Communication Networks**]: Internetworking

General Terms

Design, Economics

Keywords

Internet, Net-Neutrality, Auction

1. INTRODUCTION

> *Keeping the net open and transparent for new applications is the most important goal.*
>
> – Clark et al. [10]

The success of the Internet is largely due to its openness [10]. Openness fosters innovation. New applications can attach to the Internet without the permission of the network, and users can access any content or application at will. It is the myriad Internet applications that generate new value, improve efficiency, and advanced the societal life into an information era. A stark contrast is the plain old telephone network, which precedes the Internet by nearly a century, but supports less than a handful of applications.

However, a recent trend raises serious concerns about the future of the Internet [21]. The CEO of a large ISP publicly announced the intention not to give Internet upstarts (e.g. Google, Yahoo!) a "free-ride" [6], and novel applications such as VoIP and BitTorrent are blocked or degraded [4, 7, 8]. Were this trend to continue, the Internet might gradually erode into a closed network and stifle future innovations.

An earlier proposal [22] calls for using techniques such as data encryption and IP anonymization to fight the trend. This approach will inevitably escalate the tussle between users and ISPs: ISPs will use more sophisticated tools such as traffic analysis to block or discriminate, and users will then deploy counter techniques such as steganography to evade those tools, and ISPs will in turn attempt to crack those techniques, and so on.

This paper tackles this problem from a different perspective. We give ISPs the benefit of doubt that their actions to block or discriminate are justifiable, as clarified in [13, 17]: bandwidth-intensive applications have created a bandwidth demand that reaches the capacity limit of ISPs' networks. Continuously upgrading their networks is too costly. Hence, when there is not enough bandwidth for everyone, it is reasonable to differentiate.

This paper proposes to let ISPs auction their bandwidth among users, instead of blocking or degrading applications without considering their values to users. A user explicitly signals how much he values a network communication by placing a bid in a packet header. ISPs prioritize packets according to their bids if congestion occurs. To prevent a user from overstating his valuation or probing the network for the minimum winning bid, ISPs charge the user the highest bid of a packet that does not receive a prioritized service. Conceptually, this is a Vickrey auction [14, 15]. Users have incentives to bid their true valuations.

This proposal has a number of advantages. First, it helps to keep the net open. It is the users that decide which packets receive prioritized services. An ISP that merely intends to address the problem of bandwidth scarcity (not exporting their market power to the application layer) does not have to decide which applications to block or degrade. If congestion occurs, applications that have low values to users are automatically slowed down, and network resources are efficiently allocated to those that value them the most. Second, ISPs do not need to be the "evil-doers" that raise prices to provide quality of service. Instead, it is the users that compete among themselves for the limited capacity of ISPs' networks. Third, it is possible that ISPs can absorb some consumer surplus from highly valued applications without content or application based differenti-

NetEcon'08, August 22, 2008, Seattle, Washington, USA.

ation. For instance, if a user wants to watch an online movie on a Saturday night but the network is congested, he may be willing to pay a few extra dollars to the network to receive high-quality video. Lastly, an auction approach allows the market clearing price to dynamically adjust to the traffic demand, addressing a key challenge of congestion pricing [15, 19].

Packet auction was previously proposed by Mackie-Mason and Varian [15], but they did not present a companion technical design. The main contribution of this work is a feasible design that enables packet auction. The design addresses a number of challenges, including how to efficiently and robustly support auctions at multiple providers, how to do practical billing and accounting, how to support sender-receiver joint bid, and how to handle packet losses. The design does not require per-packet micro-payment, and fits into the present bilateral billing model of ISPs. A preliminary analysis suggests that the design is feasible with modern hardware.

We note that in its present form, the design must be far from optimal. In addition, packet auction is a concept that has not been tested in practice. The goal of our study is to understand the feasibility of this design option. This could be the first step towards a deeper study on whether packet auction is a viable practical choice and how it compares with other possible alternatives, such as flow-level auctions. We hope our work can serve as a starting point to spark future work in this direction.

The rest of this paper is organized as follows. We discuss related work in § 2. § 3 describes the design and § 4 provides a feasibility analysis. We discuss how end systems may place bids on packets in § 5. We conclude in § 6.

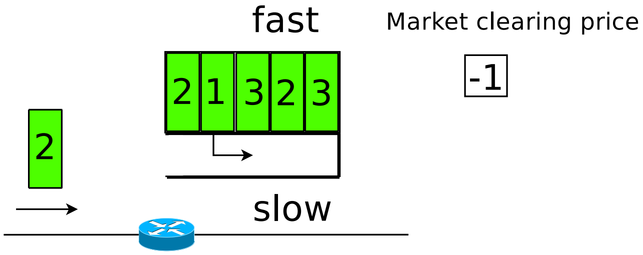
2. RELATED WORK

Mackie-Mason and Varian [15] propose to use packet auction (the *smart market*) to account for the congestion costs that one packet causes on others, but do not provide a technical design. Semret [18] analyzes different bidding strategies and their convergence properties.

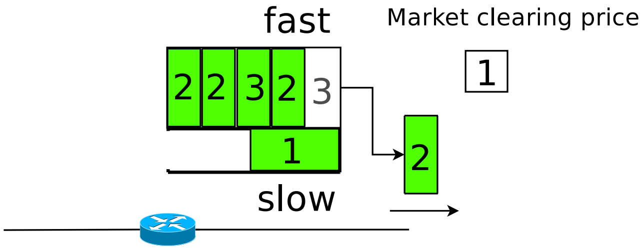
Other researchers propose to use explicit congestion prices, rather than auction, to reduce traffic demand when congestion occurs. Gibbens and Kelly [12] propose to mark flows proportional to their sending rates when congestion occurs, and charge marked packets at congestion prices. Re-ECN [5] gives senders the incentive to echo congestion markings so that they can be charged directly by their access networks. Edge pricing and expected capacity pricing [9, 19] propose to charge a user based on a capacity profile, and shape his traffic according to the profile. The M3I [1] project studies the high-level building blocks that enable flexible Internet pricing structures. The Pairs Metro Pricing scheme [16] proposes to offer different levels of priority services and charges a higher-level service a higher price.

Explicit congestion pricing has the drawback that the congestion price is difficult to set [15, 19]. This is because congestion is dynamic, and the consequence of congestion, delayed transmission, has different costs on different users and applications. This work aims to provide a technical design that enables packet auction. It complements existing work on congestion pricing, and provides the market an additional option to address the problem of resource scarcity.

Bill-pay [11] is a system that allows users to place a micropayment in a packet header to choose the desired quality of service. ISPs bill each other at the end of a billing cycle based on the micro-payment they take from each other's traffic. Our design differs from Bill-pay in two key aspects. First, in our design, users place bids, not explicit payments. The mark clearing price is automatically derived from users' bids, not set by ISPs. Second, we constrain the billing scheme to fit within the present bilateral billing model, in which adjacent ISPs pre-negotiate who-pays-whom, i.e., the customer-provider relationship, when they interconnect. In contrast, Bill-pay alters this bilateral billing scheme, and the payment between adjacent ISPs flows in both directions.



(a) **Before**



(b) **After**

Figure 1: The market clearing price is -1 when a packet with a bid 2 arrives. However, the fast lane is full. A packet with the lowest bid in the fast lane is demoted to the slow lane, and the market clearing price is updated to 1. Packets leaving the fast lane is charged at the market clearing price 1.

3. DESIGN

This section sketches a design that supports packet auction. The key challenges we address include: 1) how to implement the auction efficiently, 2) how to bill users, 3) how to do robust, efficient, and practical accounting, and 4) how to handle packet losses.

3.1 Packet Auction With Asynchronous Arrivals

A packet auction does not exactly map to a real-life auction because packets arrive asynchronously. While a packet is queued, other packets with different bid values may be discarded. At which value should the packet be charged? Moreover, how should a router handle a packet that loses its bid? If the router discards the packet immediately, flows with losing bids may be starved.

To address these issues, our design uses two queues to implement packet auction. A router maintains a fast lane and a slow lane by assigning different weights to the queues. These two queues are work-conserving. A router uses the bid in a packet header to determine which packets enter the fast lane, and a packet with a losing bid is enqueued at the slow lane to prevent starvation. For simplicity, we assume all packets are of unit size at the moment. We discuss the issue of various packet sizes in § 3.6. The highest bid among packets in the slow lane is referred to as the market clearing price.

When a packet arrives, if its bid is not above the market clearing price, it is enqueued into the slow lane. Otherwise, it is enqueued into the fast lane. If the fast lane is full, then the bid in this packet is compared against all packets in the fast lane. A packet with the lowest bid is demoted to the slow lane, and the market clearing

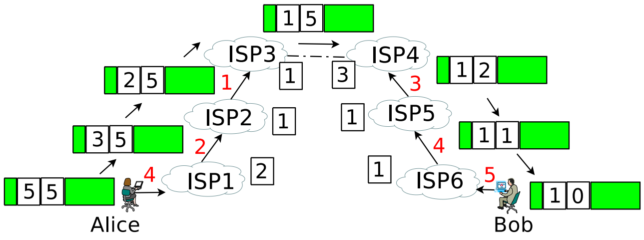


Figure 2: **A packet carries a sender bid 5 (left) and a receiver bid 5 (right). A provider bills a customer the sum of the bidding costs incurred at its network and all upper-level providers' networks.**

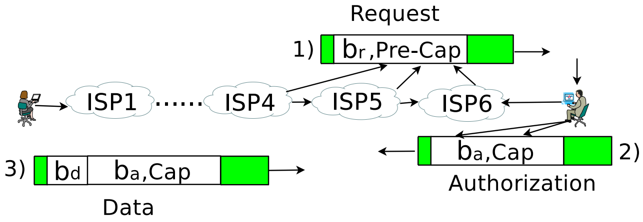


Figure 3: **A sender and a receiver negotiate a verifiable bid using extended network capabilities. Only the receiver's bid header is shown for clarity.**

price is updated to the bid of the demoted packet. If there are multiple packets with the same lowest bid, either randomness or FIFO order can be used to break the tie.

It is possible that the slow lane is full when a packet is demoted. If this demoted packet is discarded, then the market clearing price becomes inaccurate. To address this issue, our design randomly selects a packet with a bid lower than the market clearing price from the slow lane and discards that packet. The design does not further prioritize packets in the slow lane based on their bids. This is to provide flows with losing bids a roughly equal chance to get a packet through, which as we discuss later, can notify the sender to increase its bid. It is possible that a packet demoted from a fast lane arrives earlier than a packet already queued in the slow lane. To avoid further penalizing a demotion, our design inserts a demoted packet into the slow lane according to its arrival time, i.e., using the FIFO order.

A fast lane packet will be charged at the market clearing price (if it is non-negative) when it leaves. The bid in its header is decreased by the amount of the market clearing price. This process simulates a Vickrey auction among all packets that have arrived in the queues when a fast lane packet departs: the packet pays the highest bid that loses the fast lane service at that time. A packet that departs from the slow lane is not charged. But after it departs, the market clearing price is updated to the highest bid among the remaining packets. When the slow lane becomes empty, we set the market clearing price to -1. This allows a packet with a bid zero to enter the fast lane when the slow lane is empty.

When a packet is demoted, our design sets a demotion bit to notify the sender. This notification is similar to ECN. The sender is expected to take actions to avoid losing its packets in the future. Packets without a bid are treated as if they have a zero bid.

Figure 1 shows an example. When a packet with a bid 2 arrives, the fast lane is full, and the market clearing price is -1. The packet with the lowest bid 1 in the fast lane is demoted to the slow lane, and the market clearing price is increased to 1. When a packet with a bid 3 departs from the fast lane, it is charged at the price 1, and its bid is decreased to 2.

A packet may traverse multiple links. At each link, the packet will be charged at the market clearing price of that link if it gets into the fast lane. Our design ensures that any packet in the fast lane has a bid no less than the market clearing price. Thus, a packet will not be left with a negative bid.

We have not studied in detail how to set the weights between a fast lane and a slow lane, As the main purpose of a slow lane is to give a flow a chance to react to a losing bid, we expect that a small fraction of bandwidth, e.g. 10% of a link's bandwidth, could be sufficient.

3.2 The Billing Model

We describe how a provider may bill for the bidding cost of a customer's traffic. We constrain our design to use the existing Internet billing model to be practical. The bidding costs can be added as an additional item in a customer's bill in addition to the existing costs, e.g. usage-based fees or connection fees.

To support this billing scheme, our design separates a bid in a packet into a sender bid b_s and a receiver bid b_d. A bidding header in a packet thus includes a sender portion and a receiver portion. The sender bid b_s is used to enter auctions in the sender's providers' networks, and b_d is used for auctions in the receiver's providers' networks. For simplicity, our design does not use a separate bid for each provider on a packet's path. Instead, a sender (or a receiver) bid is the total price a sender (or a receiver) is will to pay for all its providers.

An ISP charges a customer the bidding costs incurred by the customer's traffic at itself and all its upper-level providers. A sender and a receiver is charged respectively according to the bidding costs at their provider networks.

Figure 2 shows an example of how ISPs bill their customers. ISP_3 and ISP_4 are two tier-1 providers. An arrowed line points from a customer to a provider, and the digit next to a line is the amount of charge, and the digit in a box next to an ISP is the market clearing price at each ISP. Suppose Alice sends a packet with a sender bid 5 and a receiver bid 5 to Bob, and it is charged a total of four units of payment in Alice's providers' networks, and five units of payment in Bob's providers' networks. ISP_1 charges the sender Alice the sum of the costs incurred at all upper-level providers and itself: ISP_1, ISP_2 and ISP_3, a total of four. ISP_2 charges ISP_1 two units of payment for the bidding costs of ISP_1's traffic within its network and its provider ISP_3's network. Similarly, ISP_3 charges ISP_2 one unit of payment for the bidding cost of ISP_2's traffic. A similar process happens on the receiver side.

3.3 Receiver Bid

A receiver's provider needs to verify that a receiver has authorized a bid b_d in a packet, because a receiver will be charged for this bid. This issue can be handled by extending network capabilities [23]. We briefly describe this process in Figure 3, and more details on capabilities can be found in [23]. The sender bid header is omitted for clarity. A sender first sends a request packet to request a receiver bid level b_r. ISPs stamp pre-capabilities. If its policy allows, the receiver returns an authorized bid b_a and capabilities to the sender. The sender can then send packets with a receiver bid b_d and capabilities. The capabilities specify the authorized bid b_a. A receiver's provider honors b_d if the capabilities verify and the receiver's bid b_d is no more than the authorized bid b_a. The provider uses b_d to determine whether a packet is qualified to enter the fast lane and decrease b_d correspondingly as described in § 3.1.

The first receiver's provider on the path must also verify that the original b_d equals b_a for accounting purpose (§ 3.4). As an optimization, b_a and capabilities can be cached by a receiver's providers to reduce packet header overhead [23].

3.4 Accounting

Next, we discuss how an ISP accounts for the bidding costs of a customer. The challenge is how to obtain the costs incurred by individual customers at upper-level providers.

3.4.1 The Receiver Side

For a customer's inbound traffic, a provider may account for the bidding costs by summing up the difference between the authorized receiver bid b_a and the current receiver bid b_d of a customer's traffic. This difference is the total charge of a packet by all upper-level providers and the provider itself. The provider can keep a counter C_u for each customer u. When a packet is forwarded to u, it increases C_u by the amount $(b_a - b_d)$. At the end of a billing cycle, the provider bills the customer u the amount C_u.

3.4.2 The Sender Side

The accounting problem at the sender side (i.e. a customer's outbound traffic) is complicated because the upper-level providers are downstream on the path. A sender's provider does not know the bidding costs at those providers when it receives a packet. One possibility is to ask an upper-level provider to inform its customers the bidding charges of their traffic. However, an upper-level provider must break down the charges to the granularity of individual subscribers. Otherwise, an access provider is unable to accurately bill a subscriber for the bidding cost of his traffic. This requirement may become a scalability issue at large providers.

Our design trades accuracy for scalability. It enables an access ISP to charge individual subscribers accurately for the bidding costs of their traffic, but lets ISPs bill large customers, e.g. other ISPs or organizations, based on statistical averaging. We think that large providers would have sufficient traffic aggregation to ignore cost variations over time. We describe how to obtain the average bidding cost and an individual subscriber's bidding cost respectively.

Obtain Average Bidding Costs: The high-level idea is to let an ISP obtain the average market clearing prices from its providers, and charge a customer at the average prices. We describe this process formally using a fluid model. Let $M_{P_l}(t)$ denote the market clearing price of a provider P_l at time t, and $\wp_h = P_l P_{l+1} P_{l+2} ... P_h$ denote a path that starts at the provider P_l, ends at a provider P_h, and includes only customer-provider links. Let $T^{P_l}_{u,\wp_h}(t)$ denote the rate of the traffic from a customer u to a provider P_l that is forwarded along the path $\wp_h$, and $B^{P_l}_{u,\wp_h}(t)$ be the distribution of bids of this traffic. The bidding costs $\Delta C^{P_l}_u(t)$ incurred by u's traffic at P_l and P_l's providers in an interval $[t, t + \Delta t)$ can be computed as follows:

```
1. for each path $\wp_h$: do
2.     for $i \in l, ..., h$: do
3.         if $(B^{P_l}_{u,\wp_h}(t) \geq M_{P_i}(t))$ then
4.             $\Delta C^{P_l}_u(t) += T^{P_l}_{u,\wp_h}(t) \cdot M_{P_i}(t) \cdot \Delta t$
5.             $B^{P_l}_{u,\wp_h}(t) -= M_{P_i}(t)$
```

Line 3-5 simulate how the amount of traffic $T^{P_l}_{u,\wp_h}(t) \cdot \Delta t$ with the bid $B^{P_l}_{u,\wp_h}(t)$ is charged at each provider P_i along the path $\wp_h$. The provider P_l can obtain the total bidding cost of a customer u by integrating $\Delta C^{P_l}_u(t)$ over a billing cycle: $\int \Delta C^{P_l}_u(t)$.

Our design uses the average market clearing price $\overline{M_{P_i}(t)}$ to approximate $M_{P_i}(t)$, as a provider P_l does not know the instantaneous value $M_{P_i}(t)$ of a downstream provider P_i. Note that an average market clearing price can be automatically determined from the bids carried by packets (See § 3.1). A provider P_l keeps a histogram of u's traffic binned by its bids and paths over an interval Δt. Let $\Delta T^{P_l}_{u,\wp_h,b}(t)$ be the amount of traffic in a bin of the bid value b. Each provider P_i sends a customer its average market clearing price $\overline{M_{P_i}(t)}$ averaged over Δt, and the customer propagates this value to its customers and so on. The provider P_l uses the traffic histogram and the average market clearing prices from its upper-level providers to approximate the computations in Line 3-5 as follows:

```
6.     for each $b \geq \overline{M_{P_i}(t)}$ in the ascending order: do
7.         $\Delta C^{P_l}_u(t) += \Delta T^{P_l}_{u,\wp_h,b}(t) \cdot \overline{M_{P_i}(t)}$
8.         $\Delta T^{P_l}_{u,\wp_h,(b-\overline{M_{P_i}(t)})} += \Delta T^{P_l}_{u,\wp_h,b}(t)$
9.         $\Delta T^{P_l}_{u,\wp_h,b}(t) = 0$
```

Line 8-9 simulate the case that after the amount of traffic $\Delta T^{P_l}_{u,\wp_h,b}$ with a bid b is charged the average market clearing price $\overline{M_{P_i}(t)}$, its bid becomes $b - \overline{M_{P_i}(t)}$.

The averaging interval Δt can be short for a higher accuracy, or as long as a billing cycle for simplicity. We think that it can be set to five minutes to be consistent with the commonly used 95% and 5-minute billing scheme.

Obtain a Subscriber's Bidding Cost: If a sender sends a packet with a bid b_s, and when the packet exits the sender's last provider on the path, the bid becomes b'_s, then the bidding cost for this packet is $b_s - b'_s$. The sum $\sum b_s - \sum b'_s$ over all traffic of the sender is the total bidding cost of the sender. An access provider can easily track the sum $\sum b_s$ when it receives a sender's packets. The challenge is to obtain the exit bid b'_s.

Our design lets a receiver echo back the exit bid b'_s to the sender (e.g. piggyback on TCP ACKs), and the sender submit this value to its access provider on a subsequent packet. If a sender does not return a bid b'_s, it may be charged more than its actual cost.

Our design uses a keyed one-way hash function to prevent a sender from lying about the exit bid b'_s, as shown in Figure 4. When an access provider receives a packet from a sender, it inserts a nonce n and a key k into the packet header, as shown in step 1 in Figure 4. The key k can be generated from a time-varying master key, the nonce n, and the sender's IP address, using a pseudo random number generator or a hash function. When the packet exits the sender's last provider on the path, e.g., ISP_3 in Figure 4, the exit provider computes a keyed hash over the exit bid b'_s: $hash(n, k, b'_s)$, and replaces the key with the hash value. The receiver echos back b'_s, the nonce n, and the hash value as shown in step 3 in Figure 4. The sender submits the exit bid b'_s, the nonce n, and the hash value in a subsequent packet (step 4 in Figure 4). The access provider recovers the key k from its master key, the nonce n, and the sender's IP, and verifies $hash(n, k, b'_s)$. If a sender lies about the exit bid b'_s, the hash value would not verify. The nonce n prevents a sender from submitting the same exit bid twice. The access provider changes its master key every few seconds to prevent a sender from guessing the key k using brute force search. A sender has incentives to submit its exit bid within the key expiration time, because otherwise, the hash value will not verify and a sender will be charged more.

3.5 Packet Loss

Packet loss impacts the accuracy of accounting. If a packet is discarded in the middle of the network, the receiver can not echo back the exit bid to the sender, and the sender will be charged more.

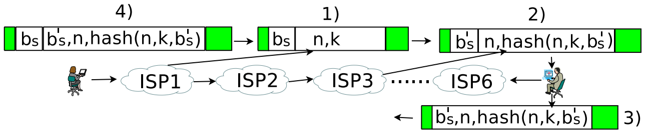


Figure 4: **An access provider uses a keyed one-way hash function to prevent a sender from modifying its exit bid. Only the sender bid header is shown for clarity.**

One possibility to improve the accuracy is to use the packet obituary framework [3] to inform an access provider of the remaining bid of a discarded packet. But this approach increases design complexity.

Instead, our design trades accuracy for simplicity. We let an access provider reduce the bidding cost of a sender $\sum b_s - \sum b'_s$ by a small amount to compensate for packet loss. If the average packet loss rate is ϵ, the access provider may reduce the sender's cost by $\epsilon * \sum b_s$. This provides incentives for a sender to keep its loss rate no higher than the average. As a packet loss wastes network resources, a well-behaved sender should always reduce its sending rate or increase its bid when the network is congested.

If a packet is discarded in one of a receiver's providers, the provider will be charged by its upper-level providers for the bidding cost of the packet, but the provider may not recover this cost from its customers. Normally, packet loss in the receiver's network will not be a significant problem, because a sender has the incentive to keep the loss rate low.

However, a problem may arise if a sender and a receiver collude to launch a DoS attack. A receiver may authorize more traffic than it can receive, wasting upstream network resources without paying for it. In addition, a colluding sender may send traffic whose TTL expires before it reaches the receiver. We note that this problem is not caused by enabling packet auction: malicious hosts can already launch this type of attack in today's Internet. Although auction makes it worse, as the colluders may send TTL-expired packets with high bids, an solution that can limit this attack in today's Internet can also limit this attack in a network that supports packet auction.

3.6 Granularity of A Bid

The unit of bid in a packet could be payment per packet, or payment per byte. Our design chooses the latter, because it simplifies bid comparison. A router compares bids in packet headers without considering the packet size, but when it charges a packet, it charges the market clearing price multiplied by the packet size. A granularity of payment per byte allows a small number of bids to represent a wide range of values as the packet size varies. For instance, if one unit of bid corresponds to one nano-dollar, and eight bits are used to represent a bid, a packet's value may range from 40 to $3.84 * 10^5$ nano-dollars, assuming 40 to 1500-byte packets.

4. FEASIBILITY ANALYSIS

We analyze the feasibility of the design by examining its overhead. The design introduces memory, processing, and packet header overhead. The memory overhead is the per-customer accounting state, which we think is negligible. The processing overhead comes from 1) the queuing scheme to support an auction, 2) the hash computation to secure a sender's exit bid, and 3) capabilities. As we only use two weighted queues: a fast lane and a slow lane, we think various queuing operations can be done efficiently. In particular, if the queue length is small [2], we expect that the processing costs of demoting the lowest bid packet from a fast lane, inserting a demoted packet to a slow lane according to the FIFO order, or selecting a random packet with a lower bid than the market clearing price to discard from a slow lane are low. Our openssl speed test shows that a hash computation derived from the block cipher AES takes 165 nano-seconds on a PC with an AMD Opteron 285 2.6GHz CPU. As each packet only requires one such computation, this CPU can perform the hash computations for six million packets every second. According to [23], the majority of capability packets have little processing overhead after capabilities are cached. If we assume CPU is the bottleneck resource, we think a prototype implementation of our design can support a few Gbps throughput, and a hardware optimized implementation may achieve an even higher throughput.

If we use an 8-bit bid, a 16-bit nonce, and a 32-bit hash value, we expect an eight-byte overhead for a sender's bid header and a nine-byte overhead for a receiver's bid header (eight bytes capabilities [23] plus one byte bid). Returning exit bids adds another seven-byte overhead, but multiple exit bids can be sent in one packet. This header overhead is non-trivial given an Ethernet MTU of 1500 bytes. If the jumbo frame size of Gigabit Ethernet (9000 bytes) becomes more popular in the future, the header overhead is less a problem. Furthermore, if a user does not wish to receive better than the best effort service, he can send packets without a bid header. This further reduces the header overhead.

5. DISCUSSION

This work is in an early stage and has left a number of issues for future work. This section discusses these open design issues.

5.1 Billing a Flat-Fee Subscriber

Bidding costs may cause a customer's monthly charge to vary. For large customers, it is unlikely a problem because they are already charged usage-based fees. However, it has been shown [20] that individual subscribers prefer a flat-fee, even if the fee is higher than usage-based fees. We discuss how an access provider may support packet auctions with a flat-fee based billing scheme.

An access provider may allocate a subscriber an amount of "free" bidding tokens to avoid unpredictable usage-based charges, similar to the peak-time minutes in cellular phone plans. Each token corresponds to one unit of bid. A higher monthly fee is allocated more monthly tokens. If a user's traffic incurs a bidding cost, the cost is taken from his token bank. If a user uses up his tokens, it is a signal that he has consumed his share of bandwidth during network congestion. The user may purchase additional tokens to receive better than the best effort service, or send traffic without a bid header to receive the best effort service.

5.2 End Host Algorithms

Host software needs to be modified to take advantage of this auction framework. We imagine that each application may have a default bid value and a user can overwrite it according to his preferences and perceived performance. A software agent will assist a user to translate the amount he is willing to pay to a bid in a packet header. For instance, if a user is willing to pay 1 cent per minute for international VoIP, the software agent may translate this amount to a per packet bid by dividing the encoding rate of VoIP.

If a user's bid is low and his traffic is demoted to the slow lane, an inelastic application should use self-administered admission control to send later, or reduce its encoding rate but increase the per-packet bid to meet a user's budget. An elastic application may use TCP-like congestion control algorithm to adapt its sending rate to the available bandwidth.

5.3 Dishonest ISPs

Our design assumes that ISPs will honestly charge the Vickrey price. Dishonest ISPs may overcharge a packet. Presently, our design relies on market competition to discipline ISPs. If a dishonest ISP overcharges, a user may notice that his network cost is high, and switch to a lower cost provider. It is within our future work to design mechanisms for a user to detect misbehaving ISPs and switch providers at a finer time scale such as at the packet level.

6. CONCLUSION

This paper argues that packet auction may be a better alternative for ISPs to address the bandwidth scarcity problem than blockage or discrimination. Users place bids in their packets. ISPs allocate bandwidth to the most valuable packets, and charge them the Vickrey auction price. We outline a design that addresses a number of challenges arising from supporting this auction. A preliminary analysis suggests that the design has reasonable overhead. This work may serve as a starting point to future efforts on designing a satisfactory solution to address the tussle between users and ISPs [10].

Acknowledgement

The author would like to thank Jon Crowcraft and Bob Briscoe for useful discussions via email exchange. She also thanks Xin Liu, Ang Li, and Michael Sirivianos for reading an early draft of this paper, and the anonymous reviewers for their useful comments. This work was supported in part by the National Science Foundation grants CNS-0627787 and CNS-0627166.

References

[1] M3I – Market Managed Multiservice Internet. `http://www.m3iproject.org/`.

[2] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing router buffers. In *ACM SIGCOMM*, Oct. 2004.

[3] K. Argyraki, P. Maniatis, D. Cheriton, and S. Shenker. Providing packet obituaries. In *ACM HotNets-III*, 2004.

[4] R. Beverly, S. Bauer, and A. Berger. The Internet's not a big truck: Toward quantifying network neutrality. In *Proc. of PAM Workshop*, Apr. 2007.

[5] B. Briscoe. Re-ECN: Adding accountability for causing congestion to TCP/IP. IETF draft, `draft-briscoe-tsvwg-re-ecn-tcp-04`, 2007.

[6] BusinessWeek. Online extra: At SBC, it's all about "scale and scope", Nov. 7, 2005.

[7] B. Charny. Vonage says its calls are still being blocked. CNET News.com, Mar. 21, 2005.

[8] J. Cheng. Evidence mounts that Comcast is targeting BitTorrent traffic. ars technica, Oct. 2007.

[9] D. Clark. A model for cost allocation and pricing in the Internet. In *MIT Workshop on Internet Economics*, 1995.

[10] D. Clark, J. Wroclawski, K. Sollins, and R. Braden. Tussle in Cyberspace: defining tomorrow's Internet. In *ACM SIGCOMM*, Aug. 2002.

[11] C. Estan, A. Akella, and S. Banerjee. Achieving good end-to-end service using bill-pay. In *ACM HotNets-V*, 2006.

[12] R. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 1999.

[13] J. Krim. Executive wants to charge for web speed. Washington Post, Dec. 1, 2005.

[14] V. Krishna. *Auction Theory*. Academic Press, 2002.

[15] J. K. MacKie-Mason and H. R. Varian. Pricing the Internet. In *Public Access to the Internet*, May 1993.

[16] A. Odlyzko. Paris metro pricing for the Internet. In *Proc. of the 1st ACM conference on Electronic Commerce*, 1999.

[17] M. Reardon. AT&T chief, FCC chair clarify on net neutrality. CNET News.com, Mar. 21, 2006.

[18] N. Semret. *Market Mechanisms for Network Resource Sharing*. PhD thesis, Columbia University, 1999.

[19] S. Shenker, D. D. Clark, D. Estrin, and S. Herzog. Pricing in computer networks: Reshaping the research agenda. *ACM CCR*, 1996.

[20] H. R. Varian. The demand for bandwidth: Evidence from the INDEX project. `http://www.ischool.berkeley.edu/~hal/Papers/brookings/brookings.html`, Feb. 2002.

[21] Network Neutrality. `http://en.wikipedia.org/wiki/Net_neutrality`.

[22] X. Yang, G. Tsudik, and X. Liu. A technical approach to net neutrality. In *ACM HotNets-V*, Nov. 2006.

[23] X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting network architecture. In *ACM SIGCOMM*, Aug. 2005.