`RABID`: Random Auctions for Bandwidth in Internet Devices

A thesis presented
by

Jeremy D. Hoon

To
Computer Science
in partial fulfillment of the honors requirements
for the degree of
Bachelor of Arts
Harvard College
Cambridge, Massachusetts

April 1, 2011

Contents

Symbols

Symbol	Domain	Value	Description
s	n/a	n/a	The seller
n	$\mathbf{N}$	n/a	Number of buyers
N	n/a	$\{1, 2, \ldots, n\}$	The set of buyers
i	N	n/a	A single buyer
u_i	$\mathbf{R}$	n/a	The utility of buyer i
v_i	$\mathbf{R}_+$	n/a	The per-byte value of data derive by buyer i
a_i	$\mathbf{N}$	n/a	The allocation (in bytes) received by buyer i.
d_i	$\mathbf{N}$	n/a	Buyer i's maximum demand (in bytes).
p_i	$\mathbf{R}$	n/a	Buyer i's per-packet price.
b_i	$[r, \infty)$	n/a	Buyer i's bid.
r	$\mathbf{R}_+$	n/a	Reserve price set by s.
b	T	$(b_1, \ldots, b_n)$	The bid vector for all buyers in N.
T	n/a	$[r, \infty)^n$	The domain of bids.
$\mathcal{A}(\cdot)$	$b \to \mathbf{N}_+^n$	n/a	The allocation rule.
$\mathcal{A}$	n/a	$\mathbf{N}^n$	The realized allocation vector.
$\tilde{\mathcal{A}}(\cdot)$	n/a	$\mathbf{N}^n$	The allocation rule transformed by Babaioff et al.'s generic transformation.
$\mathcal{O}$	n/a	$\{\mathcal{O}_{FIFO}, \mathcal{O}_{STRICT}\}$	The mechanism's induced routing prioritization.
$P_i(b)$	$T \to R_+$	n/a	Buyer i's payoff function (the payment rule).

Abstract

Mobile internet connectivity is an increasingly scarce yet indispensable commodity for modern internet users. Our work suggests a method for providing mobile internet connectivity to laptops, tablets, and other devices without wide-area network adapters. We present `RABID`, an efficient mechanism for exchanging Internet bandwidth between untrusted, self-interested agents. `RABID` addresses the unique challenges of bandwidth trading, including the lack of buyer-seller accountability and the impossibility of computing counterfactual bandwidth allocations. At the core of our mechanism are two procedures: a randomized auction mechanism due to Babaioff, Kleinberg, and Slivkins [2], and a large-scale payment redistribution method. We provide a theoretical analysis of `RABID`, and conduct detailed simulations to investigate its behavior. Our simulation results indicate that `RABID` produces more truthful, constrained-efficient outcomes than two alternative mechanisms.

Acknowledgements

I am deeply indebted to the individuals who made this thesis possible. I would like to thank my advisor, Professor David Parkes, and my advisor-in-training, Victor Shnayder.

David, your thoughtful advice and warm encouragement have proven invaluable. Somehow, you always seem to know exactly where to look for that next bit of insight or inspiration. I could not have asked for a better guide to the heretofore unknown world of mechanism design.

Victor, I know that future undergraduates will be fortunate to count you as their thesis advisor. Your focus and foresight were essential to driving this thesis forward. Your mentorship and generous assistance has been invaluable to me.

I would like to extend my thanks to my readers, Greg Morrisett and Margo Seltzer. Greg and Margo: I know that you will help make this thesis a powerful learning experience.

Thank you to the SEAS faculty, past and present, without whose instruction I could not have written this thesis.

Pamela, thank you for being my calm amid the chaos. I could not have done this without your caring and encouragement.

Finally, my warmest thanks to my family, Carole, Doug, and Charlotte, whose love and support has been essential to my success at Harvard.

Chapter 1

Introduction

The proliferation of networked devices strains all levels of today's network infrastructure. All too often, modern internet users struggle to find reasonably-priced internet access for their laptops, tablets, or other mobile internet devices. Wireless access points are ill-equipped to handle the surge in demand for connectivity that accompanies large gatherings of internet users. A recent piece in the New York Times [1] chronicles the difficulties faced by technology journalists in obtaining connectivtity during press conferences. It is only a matter of time before average internet users demand connectivity with the same insistence as today's tech press.

Congestion is only one of the barriers to connectivity faced by mobile device users. In airports and hotels, network access is often unreasonably and inflexibly priced. A business traveller seeking to check her email during a one-hour layover between flights might be charged a few dollars for only a handful of bytes.

Mobile device users need a new mode of access to the internet.

Zemilianov et al. [17] show that spilling excess demand for local-area network bandwidth onto a separate wide-area network can serve as an effective countermeasure to the spikes in demand which cripple overloaded wireless access points. However, a naive implementation of this type of load sharing requires all internet devices to provide adapaters for local- as well as wide-area networks. A more cost-effective approach is to leverage a small number of devices with both types of network adapters, such as smartphones, to provide connectivity to laptops, tablets, and other devices which offer only local-area network adapters.

The problem of diverting excess local-area network demand onto the wide-area network is then reduced to the problem of efficiently allocating smartphone users' wide-area network bandwidth. But a mechanism to allow strangers to offload bandwidth to each other could have a number of other uses. We are specifically interested in the possibility of granting internet access to mobile device users in the absence of a local area network. Using this mechanism, smartphone users could create mobile wireless hotspots for laptop and tablet users. How, then, does one design an efficient mechanism to allocate bandwidth in this situation?

Since internet users would need to make agreements between themselves rather than with a trusted wide-area network provider, this bandwidth allocation mechanism should prevent manipulation by the parties involved. All participants in the mechanism should be unwilling or unable to deviate from the intended protocol. Buyers of bandwidth (the laptop and tablet users) should be incentivized to report their true value for bandwidth. In the language of auction theory, we say that for buyers, the mechanism must be *truthful*.

We have a similar requirement for the smartphone users, who sell bandwidth. Since it is up to sellers to allocate bandwidth according to a specific protocol, we require that the mechanism be *faithful* for sellers. Faithfulness is a property formally defined by Shneidman and Parkes [10]. Informally, faithfulness is the requirement that agents find it in their best interests to follow the intended protocol established by the mechanism designers.

Buyers and sellers will only participate in a bandwidth auction if they can be confident that they will not be harmed as a result of their participation. If we guarantee the safety of our mechanism in this regard, we insist it be *individually rational*.

Finally, we cannot lose sight of the goal of *efficiency*. This means that our mechanism will give bandwidth to the internet users who get the most value from it. We discuss the properties just mentioned in greater detail in Chapter 3.

It is well known in auction theory that there exists a mechanism which is truthful for buyers, individually rational for all, and optimally efficient[1]. This is the Vickrey-Clarke-Groves (VCG) auction mechnaism. It is extraordinarily well studied and understood, versatile, and powerful. But the bandwidth trading problem imposes a number of new constraints that prove incompatible with VCG. First is the repeated nature of the mechanism. The packet-level VCG auction proposed by Varian and MacKie-Mason [14] is not truthful in a repeated setting [5]. For example, buyers might expect to get a better deal by bidding below their true value. As a result, a buyer might wait a bit longer to transmit data, but pay far less for it. This manipulability results from pricing every packet. Each time the seller routes a packet, she implicitly conducts an auction. Under other circumstances, we could extend the timeframe of this implicit auction to cover many packets. But this plan is foiled by the need to compute counterfactual information. Specifically, this redesigned auction

[1] If the seller sets a reserve price, efficiency will be sub-optimal

would need to compute the consumption of each group of $n - 1$ buyers as if the ith buyer had been absent. Unfortunately, this task is impossible because of missing information! In agent i's absence, another agent j might have received an instant message from a friend containing a hyperlink to a YouTube video. She would have clicked the link and greatly increased her consumption of bandwidth. This example illustrates that it is not generally possible to predict how agents other than i would behave in i's absence, nor is it possible to predict how much more value these agents would have received. In Section 3.3, we will present an auction method which circumvents this difficulty.

The challenges faced by mechanism designers in the networking context do not end with this loss of truthfulness. Mechanism design efforts in this context are handicapped by the inability of buyers and sellers to verify that other agents have done their best to keep their promises. For example, if a seller promises to download one megabyte from `http://example.com` on behalf of some buyer and the seller fails to hold up her end of the bargain, the buyer does not know whether there was a problem with the connection to `example.com` or if the seller willfully broke her promise. Similarly, imagine that the router reserves some amount of her routing capacity for a buyer. If this buyer does not use her entire capacity, the seller does not know if there was an issue in the upstream network or if the buyer intentionally used less than her allocation. The consequence of these uncertainties is that multi-unit auction mechanisms, in which buyers report a quantity and a price to the seller, will in general be quite easy to manipulate.

Our mechanism, `RABID` ("random auctions for bandwidth in internet devices"), seeks to overcome these issues. Despite pricing packets over an extended timeframe, we show that it is truthful in expectation for buyers. This property comes in addition to universal individually rationality for sellers, individual rationality for buyers, and faithfulness for sellers. Critically, `RABID` is approximately constrained-efficient, as we can place reasonable lower bounds on the mechanism's efficiency.

1.1 Main results

In this thesis, we present `RABID`, an auction mechanism for internet bandwidth. It is designed to allow smartphone users to convert their phones into wireless hotspots and sell internet bandwidth to nearby untrusted (and untrusting) mobile device users. We show that `RABID` is...

- nearly efficient, with reasonable lower bounds on efficiency,

- truthful in expectation for buyers over extended time frames,
- faithful for sellers,
- ex-post individually rational for buyers,
- individually rational in expectation for sellers, and
- strongly budget balanced in expectation for the market infrastructure.

Our innovation is to mediate the interaction between buyers and sellers so as to decouple buyers' bids from sellers' revenues. This allows us to separately incentivize buyer truthfulness and seller faithfulness. Buyers and sellers are given the illusion of interacting with entirely different mechanisms. Buyers bid truthfully because they have the illusion of participating in a second-price-like auction, while sellers will behave faithfully because of the illusion of participating in a first-price auction.

Rather than directly exchanging payment, the buyer and seller rely on a *market server* to act as an intermediary. Buyers and sellers periodically communicate with the market server to report information critical to pricing, such as buyers' bids or the amount of data transmitted in a certain period of time. The privacy and authenticity of messages is guaranteed by public-key cryptography. However, the role of cryptography in `RABID` ends here. We address further challenges through careful crafting of our mechanism's incentives.

In order to circumvent the need for computing counterfactual allocations, described above, we rely on a randomized auction mechanism developed by Babaioff, Kleinberg, and Slivkins [2]. We give a detailed summary of this work in Section 3.3.

We complement our theoretical analysis with detailed simulations. Our simulator provides a way to examine `RABID`'s theoretical properties in action, and allows us to compare it against a pair of alternatives: `FIXED` and `VMM`.

`FIXED` is, as its name implies, a fixed-price allocation mechanism in which all buyers bidding above some minimum price receive service. Packets are routed on a first-come, first-served basis. `VMM` is an adaptation of the packet-level VCG auction described by Varian and MacKie-Mason in [14]. We discuss these alternative mechanisms in greater detail in Section 5.1.

Simulations show that `RABID` produces more efficient outcomes than `FIXED`, and unlike `VMM`, incentivizes truthful behavior. Furthermore, comparison using simulation between our protocol and a naive application of Babaioff et al.'s mechanism to bandwidth allocation reveals that our protocol drastically reduces the variability in expected seller revenue.

In summary, we contribute

1. a description and theoretical analysis of `RABID`, an auction mechanism for internet bandwidth, and

2. detailed simulation results which validate `RABID`'s theoretical properties and contrast its performance with several alternative mechanisms.

1.2 Prior work

`RABID` addresses a problem in algorithmic mechanism design, a discipline concerned with constraining and incentivizing self-interested agents to adhere to a designer's protocol. In a 2004 paper [11], and in several related papers ([12], [10]), Shneidman and Parkes introduce and illustate the concept of *faithfulness*. This property concerns agents' externally visible actions. Since the internal workings of an agent cannot be observed, and therefore cannot be regulated, the mechanism designer must be concerned with incentivizing and constraining agents' external actions. We can think of internal actions such as computations and deliberations, while external actions produce signals observable to other agents. External actions inlcude submitting bids, allocating goods, and disbursing payment. A mechanism is faithful if the seller's only rational course of action is to follow the protocol established by the mechanism designer. Shneidman and Parkes provide a set of tools for formally establishing (or disproving) mechanisms' faithfulness. While proving seller faithfulness in `RABID` does not require these proof techniques, we will nonetheless make heavy use of the concept of faithfulness itself.

Computer scientists first faced the problem of overwhelming network congestion in the early 1990's when the National Science Foundation announced it would stop funding the embryonic internet's backbone. Not long thereafter, Mackie-Mason et al. proposed a packet-level VCG auction to allocate access to the network [15]. They observe that "If the network is not saturated the incremental cost of sending additional packets is essentially zero."[2]. This cost rises once the network cannot simultaneously accomodate the needs of all of its users. The VCG auction proposed by MacKie-Mason et. al would achieve this condition while allowing the buyers of network bandwidth to bid truthfully. However, it is also vulnerable to manipulation, as the repeated pricing of packets encourages buyers to under-bid [5].

Because `RABID`'s design places heavy emphasis on overcoming this issue, we illustrate with an example the lack of truthfulness in repetition of Varian and MacKie-Mason's mechanism.

[2] [15], pg. 5.

Imagine two buyers. The first demands eleven packets, and has a value of four cents per packet. The second buyer only demands ten packets, but has a value of five cents per packet. Both buyers bid truthfully. Now imagine that the seller can only route twenty packets at a time without dropping a packet. That is, packets arrive over a short period of time, and the seller keeps the twenty most valuable. Then, it instantly routes all twenty packets. The price charged for every packet is equal to the highest bid of any buyer whose packets were dropped. Under this model, one of the first buyer's packets will be dropped! As a result the second buyer, who has the higher value per packet, will pay four cents per packet rather than zero (or the reserve price if one exists). The second buyer would have been better off bidding three cents per packet, receiving one less packet, but paying much less overall. We discuss additional issues with VCG-based mechanisms in Section 2.2.3.

Buyer configuration

Buyer	**Demand**	**Value**
1	11	4
2	10	5

Outcomes for Buyer 2

Bid	**Truthful?**	**Packets**	**¢/packet**	**Utility**
3	No	9	0	45
5	Yes	10	4	10

Figure 1.1: Allocation and payment for two buyers under the VCG mechanism for network traffic proposed by Varian and MacKie-Mason [15]. The seller has a capacity of 20 and must drop a packet. Buyer 1 achieves higher utility by manipulating.

Zhong et al. illustrate the difficulties in engineering incentive-compatible networking protocols [18]. The authors present a protocol, called `SPRITE`, for routing packets between self-interested nodes in an ad-hoc network. The nodes must be incentivized through cleverly designed payment rules to adhere to the network protocol. Zhong et al.'s protocol describes several concepts which we employ in `RABID`. `SPRITE` requires a non-strategic agent to manage payment between nodes. Zhong et al. refer to this as a "credit clearance service," or CCS. `RABID` employs a similar concept, which we call a *market server*. Additionally, `SPRITE`'s *receipts*, which are cryptographically signed acknowledgements of data transfer, appear in almost identical form in this thesis.

In their proof of correctness, Zhong et al. address a number of attacks on simpler protocols, and show how `SPRITE` eliminates these vulnerabilities. Because we consider only a very specific family of network topologies, in which the bandwidth seller is directly connected each buyer, we avoid many of the difficulties described by Zhong et al. However, we must engage in a similar, systematic analysis of the vulnerabilities of our protocol to buyer and seller manipulations. While `SPRITE` is a valuable inspiration for `RABID`, it is not a directly applicable solution to the bandwidth trading problem because it relies on knowledge of the self-interested nodes costs for transmitting data. `SPRITE` is in a sense a fixed-price mechanism, whereas `RABID` must necessarily allow flexible pricing in order to allocate bandwidth efficiently.

Biczok et al. [4] and Mazloumian et al. [7] demonstrate how internet users can effectively cooperate to provide internet access points. Biczok et al. focus on sharing internet connectivity through immobile wireless routers attached to wired networks, rather than on mobile devices. However, these papers bring to light the tension which can arise between individual internet users who provide connectivity to others, and telecom providers who compete with them. The authors examine the incentives on users and telecom providers when users have the possibility of providing others with access to the internet. They show that the interplay of these incentives can be quite rich. While in this thesis we do not consider the relationship between users and telecom providers, in future work it may be important to consider how we can align the interests of both parties to enable efficient access to the network.

In a 2010 paper [3], Bachrach investigates two collusion schemes in VCG auctions which enable buyers to greatly increase their utility at the expense of seller revenue. These schemes, while not immediately applicable to `RABID`, serve as warning that collusion could be devastating for seller revenue. Interestingly the allocation resulting from collusion would remain efficient, and the mechanism would still be individually rational for all agents. Even though these properties are maintained in the face of collusion, there is something unsettling about the prospect of buyers cheating a seller. Thankfully, Bachrach's collusion schemes require trust between the colluders. Our analysis of `RABID` assumes that collusion does not occur due to lack of trust and technical difficulty. We will revisit this assumption in the discussion.

`RABID` relies on the work of Babaioff, Slivkins, and Kleinberg, presented in a 2010 paper [2], to compute buyers' payments. This work, on which we elaborate substantially in Section 3.3, is originally analyzed in the context of the multi-armed bandit problem, which is itself a theoretical model for online display advertising. In this context, an online advertiser (for example, a media company such as the New York Times) sells banner advertisements to its advertising customers on a per-click basis. Babaioff et al. use their mechanism transformation to apply high-performance algorithms for selecting which ads to show while simultaneously incentivizing advertising customers to bid truthfully. In this thesis, we will show that the authors' mechanism transformation has an equally powerful application to bandwidth trading problem.

1.3 Outline

In Chapter 2, we provide further motivation for `RABID` by describing a user scenario. We then discuss in detail the unique challenges addressed by `RABID`.

Chapter 3 provides background in preparation for our formal description of `RABID`. It defines key properties and terms used throughout the rest of the thesis, and introduces the randomized procedure for computing payments which is integral to `RABID`.

In Chapter 4, we give a formal description of `RABID`.

Chapter 5 contains our theoretical analysis of `RABID`.

Chapter 6 describes our simulator, and presents simulation results in support of the theoretical properties proven in Chapter 5.

We conclude this thesis in Chapter 7 with a discussion of issues raised by `RABID` and opportunities for future work.

Chapter 2

Challenges

Before exploring the details of `RABID`, we must first situate it in the context of the problem it addresses. In this chapter, we give an informal yet thorough view of the context for which we have designed `RABID`. This context is riddled with challenges against which existing mechanisms prove inadequate.

We will begin this chapter with a user scenario describing how users could employ our mechanism to access the internet. Then, we will describe the most potent challenges our mechanism seeks to overcome, and attempt to convey why providing this internet access is a difficult problem.

2.1 User scenario

Consider a scenario in which a business traveller, Alice, finds herself waiting at the gate for her flight from LaGuardia to San Francsico International. She checks her Blackberry and notices that her colleague has asked her to comment on a slide deck for a presentation he is planning to give in the next hour. Since this computing task is too complex for her phone, Alice opens up her laptop. She attempts to connect to the internet to download her colleague's message, but is stunned to discover that LaGuardia charges several dollars for wifi connectivity. In order to send and receive just a few kilobytes, she would be charged several orders of magnitude more than for transmitting the same amount of data on a mobile data network.[1]

[1] According to airportwifiguide.com, 24 hours of wifi access at LaGuardia costs $6.95. `http://www.airportwifiguide.com/lga-laguardia-airport/`, accessed March 29, 2011.

Appalled, Alice opens an application called NetMarket[2]. This application allows her to find nearby users offering to sell internet bandwidth through their phones.

Meanwhile, Bob, another traveller, is sitting nearby with his Android-powered phone in his pocket. This phone runs a version of NetMarket for Android that can route wifi traffic through the phone's wide-area network adapter. The phone presents itself as a wifi hotspot, allowing Alice to connect to the phone and, through it, the internet.

Because Alice and Bob do not know each other, and much less trust each other, Alice must rely on NetMarket to safely mediate the interaction. This is where we begin to encounter the difficulties addressed by `RABID`. In Section 2.2, we'll examine the mechanism design challenges inherent in facilitating this interaction between Alice and Bob.

2.2 Bandwidth trading challenges

2.2.1 Secure payment

Alice must pay for the bandwidth she obtains from Bob. If she were transacting directly with him, payment could be achieved with an anonymous payment method such as BitCoin [9]. But as we will see later, our mechanism requires that we decouple buyer and seller payments. In `RABID`, payment is instead rendered indirectly with the help of an independent, trusted entity known as the *market server*. In order to pay the seller for bandwidth, a buyer sends a message to the market server to notify it that the seller has provided her with a certain amount of bandwidth. We call this message a *receipt*. Over time, the market server can aggregate receipts and compute the payment owed by the buyer.

In our user scenario, Alice downloads the slide deck from her coworker. During the download, she notifies the market server every ten seconds that Bob has provided her with several hundred kilobytes. Once the download is complete and she busies herself with editing the slide deck, she informs the market server that she has not recently received any data. In no case does she actually pay Bob directly or ask the market server to pay Bob a certain amount of money. That is up to the market server to decide.

Without imposing any additional restrictions, this payment scheme is highly vulnerable to manipulation. This is because buyers and sellers are not constrained or incentivized to cooperate in reporting the true amount of bandwidth provided by the seller to the buyers. Consider that Alice relies on Bob to forward her messages to the market server. This makes

[2]This application is fictional, but is meant to demonstrate how an implementation of `RABID` might operate in practice.

it very easy for Bob to spoof packets which appear to come from Alice. He can claim to have provided tremendous amounts of data to Alice, all without Alice's knowledge.

Cryptography offers an elegant solution to this issue. Under `RABID`, buyers maintain a public-private key pair, and register their public keys with the market server. To keep Bob from forging receipts, Alice cryptographically signs her receipts with her private key. She then forwards her receipts to Bob, who examines them and verifies that the amount of bandwidth he provided to Alice is in fact the amount of bandwidth she claims to have received. Finally, Bob forwards the receipt to the market server.

If Bob notices that Alice claims to receive less bandwidth than the amount he has provided to her, he terminates his connection with Alice. We assume this threat is credible.

Unfortunately, this may not be a strong enough disincentive to Alice to keep her from cheating. Imagine that receipts are forwarded to the market server only once an hour. If Alice only needs access to the network for five minutes to check her email, she could be part way to San Francisco before she is due to report payment to the market server. As a result, Bob would never be compensated for the bandwidth he provided to Alice.

Rapidly generating receipts is critical to ensuring the functionality of `RABID`. We recommend that receipts be exchanged regularly enough to bound the seller's loss of revenue at some safe level, such as a penny, a dime, or a dollar.

We will assume that these modifications allow buyers to safely and reliably send indirect payment. However, we will see that this is not enough to guarantee a bandwidth auction which is truthful for buyers and a faithful for sellers. Achieving these two goals is the main of focus of this thesis.

2.2.2 Lack of accountability

In many auction mechanisms, buyers specify a quantity of goods or a number of items they would like to obtain. The seller can then allocate a fixed amount of these goods to buyers. Consider an auction for a commodity, such as grain. The seller may already have possession of the grain (say, ten bushels), or may be able to guarantee its delivery. Imagine that two buyers participate in the auction. The first buyer offers to buy seven bushels at twenty cents per bushel. The second buyer offers to buy ten bushels at five cents per bushel. The first buyer gets his seven bushels, and the second buyer gets the remaining three. There is no question that the seller can provide the grain, nor that the buyer can take delivery of the grain. What if we were to apply this idea to bandwidth? Buyers could submit to the seller a number of packets demanded and a price per packet. The seller could then carve up her available bandwidth between the buyers.

Unfortunately, bandwidth and grain cannot be auctioned in the same way. If Bob offers to sell Alice ten megabytes of bandwidth, he is making a promise he may not be able to keep. Bandwidth is not a tangible good that can be stockpiled. Its delivery cannot be guaranteed. Imagine that Alice is trying to read her email. Unknown to her, the system administrator at her office brought a cup of coffee into the machine room and spilled it on the mail exchange server. Bob will contact Alice's mail server and get no response. The ten megabytes of email attachments he promised Alice will not be delivered. Alice's email server could be unreachable for an infinite number of reasons unknowable to either Bob or Alice. For this reason, Bob cannot be sure that he can keep his promises to deliver bandwidth to Alice.

But the internet is generally reliable. System administrators rarely spill coffee on expensive machines. If Bob promises Alice to deliver her email, Alice should feel fairly confident that Bob is likely to be able to succeed. Why not simply expect Bob to do his best, and accept his failure to deliver promised bandwidth as an unavoidable failure somewhere upstream? Unfortunately, Alice must expect Bob to behave rationally. Since she has no means of distinguishing a network failure from Bob's intentional choice not to retrieve her data, Bob will have an incentive to cheat. He can promise to deliver an amount of bandwidth beyond his capacity to Alice and a number of other buyers, collect payment from these buyers, and then attribute his failure to deliver to a network failure.

Now imagine that instead of paying up front, buyers pay per packet delivered, but still demand a certain quantity of bandwidth. Perhaps Alice is unsure of the size of her coworker's slide deck. Under this scheme, she has an incentive to ask Bob for a large amount of bandwidth (for example, ten megabytes) even if the true size of the slide deck is only a few hundred kilobytes. Bob must then reserve this capacity for Alice, even though she will never use it. If he had been able to allocate this capacity to another buyer, Bob could have achieved greater revenue. This is a lack of buyer accountability. The seller has no way to determine if the buyer knew she would not use all of her reserved bandwidth, or if upstream network issues prevented her from doing so.

Because neither party can hold the other accountable for delivering or using a certain amount of bandwidth, we must pursue an approach which does not allocate bandwidth a priori and instead allocates traffic on a packet-by-packet basis. We show that allocating packets based on *priority* rather than based on reservations can achieve this goal.

2.2.3 Impossibility of computing counterfactuals

In Section 1.2, we referred to work by Varian and MacKie-Mason which performs allocation based on priority, as discussed at the end of Section 2.2.2. As we have mentioned, this mechanism does not provide strong enough truthfulness properties for our bandwidth exchange problem. In Section 1.2, we demonstrated how Varian and MacKie-Mason's mechanism could be gained. We will now discuss a natural response to this issue, and explain why this response also fails to address the challenges we face.

The mechanism proposed by Varian and MacKie-Mason fails because it repeatedly prices individual packets, leading to a loss of truthfulness. It is well known that VCG is not truthful in repeated settings [5]. Why not price many packets instead of just one? VCG dictates that the cost for a bundle of packets would be a function of the reported value denied to other buyers in the course of routing that bundle. At first glance, it seems simple to compute this value. All we need to do is look at the amount of value each other buyer would have received if that bundle of packets never existed. Then we just take the difference between these *counterfactual* values and the realized buyer values to determine the correct VCG payment.

The critical problem with the approach described above is that these values are counterfactual – they are answers to the question "what if?" Multiunit VCG mechanisms, in which buyers bid a quantity demanded and a per-unit price, allow mechanism designers to answer this question easily. But as discussed in Section 2.2.2, we need to allocate on a packet-by-packet basis. In order to answer this "what if?" question, we would need to rewind time and re-run the allocation mechanism while ignoring the bundle of packets we seek to price. We alluded to this issue in Chapter 1, with the example of YouTube video that a certain buyer would have downloaded if only a single additional packet could have been allocated to her. Another way to think about the impossibility of computer counterfactuals is that the seller has no idea when packets will arrive, nor does she know how many packets to expect. She only knows what to do once she has a packet to route. To the seller, the processes generating packets are inscrutable.

For this reason, `RABID` must never need to compute counterfactual allocations. In order to satisfy this constraint, we leverage work by Babaioff, Kleinberg, and Slivkins, described in Section 3.3.

Problem	Affects fixed price?	Affects Packet VCG?	Affects Multi-packet VCG?	Affects Multiunit VCG?	Affects `RABID`?
Agents cannot hold each other accountable to deliver on promises	No	No	No	Yes	No
Not truthful in bandwidth setting	No	Yes	No	No	No
Computing counterfactuals is impossible	No	No	Yes	No	No
Buyers can collude against the seller	No	Yes	Yes	Yes	Yes
Not efficient	Yes	No	No	No	No

Figure 2.1: Challenge summary. This table compares the vulnerability of candidate mechanisms to various issues.

2.2.4 Collusion

In many auctions, buyers can improve their individual and collective utilities by banding together and colluding against the seller. In Section 1.2, we alluded to work by Bachrach which describes several powerful collusion schemes to which `RABID` is vulnerable. Additionally, we have devised new collusion schemes which specifically target our own mechanism. The issue of collusion is a thorny one. As mentioned in Section 1.2, there are barriers to collusion, such as trust and communication, that make it somewhat difficult. We rely on this rationale for assuming that agents will not collude. We return to this issue in Section 7.2.

Chapter 3

Background

In Chapter 2, we motivated the problem of bandwidth exchange in a realistic user scenario. Yet we also discovered that VCG, even with the assistance of cryptography, is unsuitable in this context.

In this chapter, we will present some preliminary concepts and additional motiviation for our mechanism. We will begin by examining the desirable properties we wish to achieve in `RABID`. Then in Section 3.2, we will discuss several concepts that will be central to the discussion of our mechanism. Finally in Section 3.3, we will introduces the work of Babaioff, Kleinberg, and Slivkins in building truthful mechanisms from allocation rules. This method allows us to achieve truthful pricing of data in `RABID`.

3.1 Desirable properties of auctions

Fundamentally, we are interested in an *efficient* mechanism, by which we mean that the total utility of the mechanism's participants is maximized, or at least improved compared to reasonable alternatives. However, we will be willing to sacrifice some efficiency in order to allow the seller to achieve greater revenue. In order to allow for this trade-off, we let sellers set a *reserve price*. In order to precisely define the property we would like to achieve, we introduce the concept of constrained efficiency:

Definition 3.1.1. A mechanism is *constrained-efficient* if there exists a reserve price such that the mechanism is efficient at this reserve price.

Remark 3.1.1. For most mechanisms (including `RABID`), $r = 0$ is the reserve price which maximizes efficiency.

Next, the mechanism should be *individually rational*, meaning that no participant receives negative utility from participating. Individual rationality will have different implications for buyers and sellers, and in fact, buyers and sellers will have different guarantees of individual rationality. For this reason, we will be concerned with two slightly different properties. The first is universal ex-post individual rationality:

Definition 3.1.2. A mechanism is *universal ex-post individually rational* with respect to some agent if the mechanism guarantees this agent that it will achieve non-negative utility by participating in the mechanism.

Remark 3.1.2. In `RABID`, universal ex-post individual rationality for buyers guarantees that all buyers achieve non-negative utility for all bid vectors and across all randomness in the mechanism.

We are also interested in a weaker type of individual rationality, which holds only in expectation.

Definition 3.1.3. A mechanism which is *individually rational in expectation* for some agent yields a non-negative expected utility to this agent. This expectation is over all randomness affecting the mechanism.

We show in the sequel that (given assumptions about seller behavior), buyers under `RABID` achieve universal ex-post individual rationality, while sellers achieve individual rationality in expectation.

We would prefer to have all payments flow between buyers and sellers, rather than have to remove or inject money into the system.

Definition 3.1.4. A mechanism is *strongly budget balanced* if the sum of payments made by all agents is equal to the sum of all payments received by all agents. No money is added to or removed from the mechanism.

We say that a mechanism is only *weakly budget balanced* if cash is allowed to flow out of the system. We note that it is possible for either of these concepts of budget balance to hold only in expectation.

`RABID` relies on sellers following a specific protocol. Deviation from this protocol would lead to undesirable behavior. Sellers should therefore be incentivized to follow this protocol.

Definition 3.1.5. A *faithful* mechanism (with respect to some agent) maximizes the utility of this agent when her externally observable actions correspond to those specified by the mechanism designer's intended protocol for this agent.

While faithfulness captures our notion of adherence to protocol in sellers, we will be interested in a slightly different notion of good behavior in buyers. We are interested in encouraging buyers to bid their true values for data.

Definition 3.1.6. A buyer bids *truthfully* if she reports her true value for data to the seller. We say that a mechanism is *truthful* if buyers maximize their individual utilities by bidding truthfully. A mechanism is *truthful in expectation* if buyers maximize their expected utilities by bidding truthfully.

Truthful mechanisms obviate the need for strategic bidding. Avoiding strategic bidding saves buyers the cognitive effort of determining their optimal bid given their true value. In practice, truthfulness may also avoid inefficiencies by eliminating the convergence phase that many non-truthful mechanisms require in order to achieve a game-theoretic equilibrium between the buyers.

We show that `RABID` achieves

- constrained-efficiency,
- universal ex-post individual rationality of buyers,
- individual rationality in expectation for sellers,
- truthfulness in expectation for buyers,
- faithfulness for sellers, and
- system-wide strong budget balance in expectation.

3.2 Auction terminology

To facilitate our discussion of `RABID`, we define some useful terms as they pertain to the problem of bandwidth trading.

Auctions are inherently about assigning resources to agents. We refer to this assignment of resources as an *allocation*. In the bandwidth trading context, this terms takes on a more specific meaning.

Definition 3.2.1. A buyer's bandwidth *allocation* is the number of packets she receives over some period of time through her participation in a bandwidth auction mechanism.

In the bandwidth trading context, allocations result from the seller's packet routing behavior. For this reason, we use the terms *route* and *allocate* interchangeably.

Most traditional auction mechanisms can be conceptually decomposed into an *allocation rule* and a *payment rule*. Informally speaking, the allocation rule determines who gets the goods, and the payment rule sets the price on those goods. In the bandwidth trading context, the goods in question are network packets. The boundary between payment- and allocation rules is fluid, and the two rules tend to be highly interdependent.

Definition 3.2.2. A bandwidth auction mechanism's *allocation rule* is a procedure for determining whether a packet will be transmitted or dropped.

Definition 3.2.3. A bandwidth auction mechanism's *payment rule* sets a price for the allocation of packets received by a buyer.

In `RABID`, we perform a certain repeated process over and over. Each time, buyers submit bids, sellers route packets, and buyers pay for their allocation. We refer to each of these rounds as an *epoch*.

Definition 3.2.4. An *epoch* is a period of time in which:

- each buyer submits her bid to her associated seller,
- sellers route bandwidth to buyers based on these bids, and
- buyers are charged for the packets routed on their behalf.

A common tool for increasing seller revenue in many auctions mechanism is the *reserve price*. We alluded to this concept in Section 3.1, in our discussion of constrained efficiency.

Definition 3.2.5. A *reserve price* is the minimum bid a seller will accept. Buyers will be charged at least the reserve price for each packet they transmit.

3.3 Constructing truthful mechanisms

We have previously discussed the need for a mechanism which does not require computing counterfactual allocations. Babaioff, Kleinberg, and Slivkins [2] propose such a mechanism. They present a method for transforming any single-parameter monotone allocation rule into a truthful auction mechanism. This transformation randomly samples the payments which will incentivize buyers to bid truthfully. Crucially, the transformation requires the allocation rule to be run only once. This allows us to circumvent the need for counterfactual information, as previously alluded to in Section 2.2.3.

3.3.1 Transformation preconditions

Let's examine the properties stipulated on allocation rules by the transformation. First, the tranformation is only applicable to *single parameter* rules. This means that each buyer need submit only a single number to the seller in order for the seller to determine her allocation.

Next, the rule must be *monotone*. In the context of our user scenario, this means that, all other things being equal, Bob will never decrease the amount of bandwidth he allocates to Alice if she raises her bid.

Definition 3.3.1. An allocation rule $\mathcal{A}(\cdot)$ is *monotone* if for all $i \in N$ and for all $b_{-i} \in b$, $b_i \geq b'_i$ implies that $\mathcal{A}_i(b_i, b_{-i}) \geq \mathcal{A}_i(b'_i, b_{-i})$.

We will show that `RABID`'s allocation rule satisfies both these preconditions.

3.3.2 Myerson characterization of truthfulness

In his work on revenue-optimal auctions [8], Myseron shows that truthful payment rules for single-parameter allocation rules are characterized by

$$P_i(b) = b_i \cdot \mathcal{A}_i(b_i, b_{-i}) - \int_{-\infty}^{b_i} \mathcal{A}_i(u, b_{-i})du \tag{3.1}$$

where $P_i(b)$ is the payment owed by buyer i when the buyers' collective bid vector is b. Since `RABID` uses a single-parameter allocation rule, it seems all we need to do is compute the integral in Equation 3.1. For convenience, let

$$H_i(b) = \int_{-\infty}^{b_i} \mathcal{A}_i(u, b_{-i})du \tag{3.2}$$

But computing $H(b_i)$ is essentially impossible. Imagine performing this integration numerically. We would evaluate $\mathcal{A}(u, b_{-i})$ for many values of $u \neq b_i$. By definition, this would require counterfactual information we cannot obtain ($\mathcal{A}_i(u, b_{-i})$ corresponds to an allocation which has not necessarily occurred).

Babaioff et al.'s method sidesteps this issue by taking random samples from a distribution whose expected value is equal to $H_i(b)$. The method is therefore truthful only *in expectation*.

3.3.3 Estimating $H_i(b)$

The core of this sampling technique comes from a method for estimating the integral of a function. Babaioff et al. describe this method, which we summarize here.

We wish to estimate the integral of $\mathcal{A}_i(u, b_{-i})$. Consider that this function (of u) is defined on $\mathbf{R}_+$, since we can assume that buyers have non-negative value for packets. We are therefore interested in integrating over $I = [0, b_i]$. Since $\mathcal{A}_i(u, b_{-i})$ is a monotone increasing function, the endpoints of I will be extrema of $\mathcal{A}_i$ over I.

For this reason, we can construct CDF-like monotone function $F(z)$ defined on I such that $F(0) = 0$ and $F(b_i) = 1$. We can think of F as defining a random variable Y_{b_i}. Since Y_{b_i}'s CDF is given by $F(z)$, its PDF is given by $F' = \frac{d}{dz}F(z)$. Remember that the definition of expected value is given by

$$\mathbf{E}_{Y_{b_i}}[h(Y_{b_i})] = \int_0^{b_i} h(u)F'(u)du \tag{3.3}$$

All we need to do is cancel out the $F'(u)$ in the integrand through a judicious choice of $h(z)$. If we let $h(z) = \mathcal{A}_i(z, b_{-i})/F'(z)$ for fixed b_{-i}, we obtain

$$\mathbf{E}_{Y_{b_i}}\left[\frac{\mathcal{A}_i(Y_{b_i}, b_{-i})}{F'(Y_{b_i})}\right] = \int_0^{b_i} \mathcal{A}_i(u, b_{-i})du \tag{3.4}$$

3.3.4 Self-resampling procedures

In order to estimate $H_i(b)$, we will need to evaluate the allocation rule we are transforming with a randomly sampled value of Y_{b_i}. Then we can substitute the resulting allocation and realization of Y_{b_i} into Equation 3.4 and obtain truthful payment. Unfortunately, this creates a chicken and egg problem.

Let's say we want to randomly sample $H_i(b)$. We decide to evaluate our allocation function $\mathcal{A}(b_i)$ at a value sampled from Y_{b_i}. This means that buyers' allocations are now computed according to a new function $\mathcal{A}'_i(b) = \mathcal{A}_i(Y_{b_i})$. Therefore, we actually need to compute the truthful payments for $\mathcal{A}'_i(b)$ instead of for $\mathcal{A}_i(b)$. So, we just sample Y_{b_i} again. This creates another new allocation function, $\mathcal{A}''_i(b)$. But now we need to estimate payments for $\mathcal{A}''_i(b)$, so we create $\mathcal{A}'''_i(b)\ldots$

In order to avoid recursing infinitely, Babaioff et al. introduce the concept of a *self-resampling procedure*. They describe this concept as a "fixed point" of the recursive sampling process described above. In Section of 3.2 of their 2010 paper, the authors define the formal properties of self-resampling procedures. Their key insight is to generate two values, x_i and y_i. With high probability, $x_i = y_i = b_i$. But with small probability, the values are *resampled* below b_i. The resampled value of $y_i = u$ is chosen first. Then, the resampled x_i is chosen from a distribution equal to x_i's unconditional distribution for $b_i = u$. This allows Babaioff et al. to apply Equation 3.4 without recursing infinitely.

3.3.5 `RABID`'s self-resampling procedure

Babaioff et al. show that given a self-resampling procedure G, it is possible to construct a mechanism which is truthful in expectation from any allocation rule $\mathcal{A}(\cdot)$ which satisfies the preconditions given in Section 3.3.1. `RABID` uses a self-resampling procedure $G(b_i, r, \mu)$, given below. It adapts a procedure presented by Babaioff et al.

$$G(b_i, r, \mu) = \begin{cases} b_i & \text{with probability} \quad (1-\mu) \\ (b_i - r)\gamma^{1/(1-\mu)} + r & \text{with probability} \quad \mu \end{cases} \tag{3.5}$$

where $\gamma \sim U(0, 1)$. The parameter $\mu \in (0, 1)$ is set by the mechanism designer. We describe Its effect in Section 6.3.3. As described previously, r is a *reserve price* determined by the seller.

3.3.6 Babaioff et al.'s generic transformation

We now outline the complete procedure outlined by Babaioff et al. for building a mechanism which is truthful in expectation from a monotone single-parameter allocation rule. They refer to this procedure as the *generic transformation.* It is generic due to its compatibility with any self-resampling procedure. Since we will be using the self-resampling procedure $G(b_i, r, \mu)$ described previously in Section 3.3.4, we are interested in a specific transformation we call RESERVE.

Definition 3.3.2. Let $\mathcal{A}(b)$ be a single-parameter monotone allocation rule, mapping bid vector b to allocation vector $\mathcal{A}$. Then $\text{RESERVE}_{r,\mu}(\mathcal{A}(\cdot))$ transforms $\mathcal{A}(b)$ to a new allocation rule $\tilde{\mathcal{A}}(b)$ and a payment rule $P(\cdot)$. The transformation obeys the following procedure:

1. For each $i \in N$, obtain the resampled bid $x_i = G(b_i, r, \mu)$. Let the resampled bid vector (of all x_i) be x.

2. Allocate according to $\mathcal{A}(x)$. This yields the allocation $\tilde{\mathcal{A}}$.

3. For each $i \in N$, compute i's *rebate*

$$R_i = \begin{cases} \frac{1}{\mu}(b_i - r) \cdot \mathcal{A}_i & : \quad x_i < b_i \\ 0 & : \quad \text{otherwise} \end{cases} \tag{3.6}$$

4. For each $i \in N$, collect payment according to ch

$$P_i = b_i \mathcal{A}_i - R_i \tag{3.7}$$

5. Return the new mechansim $\mathcal{M} = (\tilde{\mathcal{A}}(b), P(\cdot))$.

We will see in Chapter 4 how $\mathcal{M}$ will form an important part of `RABID`.

Chapter 4

The `RABID` Mechanism

In this chapter, we give a detailed description of `RABID`. We begin in Section 4.1 by listing the assumptions necessary for this formal discussion. We then launch into the mechanism description proper. For clarity, we divide this discussion into two parts. This division parallels the conceptual division of `RABID` into two constituent procedures: `BKS` and `ALIGN-TRUST`. `BKS` implements bandwidth allocation and pricing according to the generic transformation introduced by the eponymous Babaioff, Kleinberg, and Slivkins, whose work we summarized in Section 3.3. This procedure is limited in scope to only a single seller and her associated buyers. For ease of explanation, we introduce machinery that will simplify the discussion of the second procedure, `ALIGN-TRUST`. We discuss `BKS` in Section 4.3

Unlike `BKS`, `ALIGN-TRUST` has a system-wide scope. Its role is to redistribute payments between sellers. We discuss `ALIGN-TRUST` in Section 4.4

4.1 Assumptions

Throughout our discussion of `RABID`, we describe the idealized version of the mechanism with which our formal analysis is concerned. However, we will periodically mention practical modification to improve the performance of the system. We predict that these changes would have only minor adverse effects on the desirable properties of the mechanism. In this section, we will outline the assumptions on which our idealized mechanism rests.

Realistically, a seller would allow buyers to connect and disconnect from her mobile device. However, we ignore this dynamic behavior, and assume that the number of buyers remains fixed. This is a reasonable assumption if we consider that we can simulate entrance and exit of buyers by modifying buyer demand. If a buyer's demand falls to zero, she will appear to have exited. Similarly, if a buyer begins transmitting after many epochs of silence, she will appear to enter.

In our analysis, we do not treat any lower-level network issues. We do not treat any link-layer behavior (Ethernet, 802.11), nor do we consider the effect of transport-layer protocols (TCP, UDP) on routing-layer (IP) packet traffic. Instead, we simply assume that packets flow effortlessly between buyers and their seller. In our model, adding buyers to the mechanism has no effect other than to increase the demand for bandwidth. We ignore any overhead from increasing the number of simultaneous local-area wireless connections to the seller, and we assume that all out-of-band communication between buyers and seller imposes no cost on either. We can justify this assumption by pointing to work by Wu et al. [16] which demonstrates methods of incentivizing wireless communication between self-interested agents. The assumption of cost-free communication between buyers and seller is especially justifiable when the local-area network is much faster than the seller's connection to the internet.

Buyers and sellers must have strong identities, in order to prevent forging of bids and receipts. This strong identity might be enforced by requiring users participating in `RABID` to register using a credit card.

We strongly prefer that buyers be unable to collude with sellers or with each other, and assume this to be the case. We discuss collusion in greater detail in Section 7.2.

We assume that the unit of allocation is one packet, rather than one byte. This makes it easy to reason about routing and pricing data. This assumption is equivalent to assuming that all packets will be of a fixed, constant size. In order to determine pricing for packets, each buyer must be able to keep track of the number of packets she has exchanged with the seller. The seller must in turn keep track of all packets exchanged with each buyer.

Finally, we assume that a single routing queue can be used to allocate and price packets. For the purpose of simplifying our analysis, we ignore the asymmetries between incoming and outgoing traffic. An implementation of `RABID` would need to maintain two separate routing queues, with separate bids.

We will discuss further assumptions relevant to specific technical details as they details arise.

4.2 `RABID`'s structure

As discussed in the chapter introduction, `RABID` can be separated into two subprocedures:

Theory	**Practice**
Payment aggregated over m epochs.	Payment aggregated over a period of weeks or months.
Single queue, with a single bid, for all network traffic.	Separate upload and download queues, on which buyers bid separately.
Set of buyers N is fixed.	Buyers periodically connect to- and disconnect from sellers.
Packet receipt transmitted to market server for each packet.	Packet receipts batched and transmitted together at some regular interval, in order to conserve bandwidth.
Packets flows are determined entirely by buyer demand and behavior at the routing layers.	Ensuring priority allocation might require modifying device behavior at the link layer or transport layer (for example, re-writing TCP packets' window size).
Buyers and sellers have a strong identity.	Participating in the mechanism requires providing credit card.
Buyers are unable to collude with each other or with the seller.	Mechanism operators institute complex systems for detecting collusion or hope that the technical complexity of collusion will agents from colluding.
All packets priced identically.	Packets priced as a function of their size.
A buyer cannot detect that she will receive a rebate in a given epoch.	Buyers may try to detect if they will receive a rebate.
Buyer utility is quasi-linear. Buyers have no budget constraints.	Buyers may have limited budgets.

Figure 4.1: Theoretical constraints and suggestions for relaxing these constraints in practice.

1. A local auction mechanism we call `BKS`.
2. A system-wide utility redistribution procedure called `ALIGN-TRUST`.

In `BKS`, we are concerned with the interaction of a single seller with several buyers. It is local in the sense that the seller and buyers in a particular instance of `BKS` are agnostic to those in other instances. In our description of `BKS`, we ignore the existence of other sellers and their associated buyers. However, we will introduce machinery which is critical to our second subprocedure, `ALIGN-TRUST`. Unfortunately, the purpose of this machinery will only become clear as we explain how it functions within `ALIGN-TRUST`.

Unlike `BKS`, `ALIGN-TRUST`, deals with large numbers of sellers, each of which independently participates in its own instance of `BKS`. `ALIGN-TRUST` re-aligns the incentives of sellers in order to incentivize seller faithfulness.

`BKS`	`ALIGN-TRUST`
Regulates interactions between one seller and a handful of buyers.	Redistributes payments among many sellers.
Incentivizes buyer truthfulness.	Incentivizes seller faithfulness.
Completely determines buyer payments.	Adjusts seller payments.
Majority of computation performed by the seller.	Majority of computation performed by the market server.

Figure 4.2: A comparison of `RABID`'s subprocedures.

The `BKS` and `ALIGN-TRUST` are subprocedures are linked by the *market server*, a trusted, non-strategic entity which communicates with buyers and sellers in order to determine their payments owed or expected. We begin our description of `RABID` with an examination of the market server. Then, we proceed with our exposition of the `BKS` and `ALIGN-TRUST` subprocedures.

4.2.1 Market server

`RABID` relies on a central coordinating entity, which we call the *market server*. In our model, the market server does not derive utility or make choices, that is to say, it is non-strategic. Instead, it simply performs as designed. In practice, this is safe assumption so long as the market server has a strong identity. For example, a market server operated by a large wireless carrier is unlikely to attempt to game buyers and sellers.

Formally, the market server is defined by the tuple (μ, R). The first member of this tuple, $\mu \in (0, 1)$ is identical in purpose to the μ defined in Section 3.3.5. It is the probability with which a buyer's bid will be resampled in each epoch. R is a finite, preferably small, set of reserve price levels. This set will have role in the formal definition of RABID's sellers and in the operation of the ALIGN-TRUST subprocedure.

The market server performs several tasks. First, it must compute buyers' payments according to Equation 3.7, such that the strategy which maximizes buyers' utility in expectation is to bid truthfully. Section 3.3.6 provides greater detail on these payments. Second, the market server determines a *tax rate* $\bar{l_s}$ on sellers. The tax rate plays a key role in linking BKS with ALIGN-TRUST. It is more fully explained in Section 4.4.

4.3 The BKS subprocedure

BKS is a repeated process which determines the interaction between a single seller and her associated buyers. Every repetition of the process constitutes an *epoch*, a concept to which we alluded in Section 3.2. In our theoretical analysis, we will assume that the process repeats exactly m times. In practice, buyers connect and disconnect from the seller from time to time, yet we feel that a fixed time-frame model reasonably approximates this dynamic. Since buyer demand is not stationary, we can simulate entering and exiting buyers.

Our description of BKS will proceed as follows:

- In Section 4.3.1, we formally define the seller and buyers who participate in the subprocedure.
- In Section 4.3.2, we list in detail each step which occurs in a single BKS epoch.
- We describe buyers' interactions with BKS in Section 4.3.3 and describe the seller's interaction with the subprocedure in Section 4.3.4.
- As a prelude to our discussion of ALIGN-TRUST, we will begin discussing how payment flows between buyers and sellers in Section 4.3.5.

Throughout our description of BKS, we ignore the existence of agents other than the single seller and her associated buyers. However, we will lay the foundation for the involvement of outside agents. When we introduce machinery whose purpose is to interface with the ALIGN-TRUST subprocedure, we will make this purpose clear in order to minimize confusion.

Player	Buyer	Seller
Identified by	i	$s = (k, r)$
Domain	$N = \{1, 2, \ldots, n\}$	$(\mathbf{N}_+, \mathbf{R}_+)$
Type	v_i	-
Utility function in one epoch	$u_i(b) = v_i\tilde{\mathcal{A}}_i(b) - P_i(b)$	$u_s(b) = \sum_{i=1}^{n}(x_i - \bar{l}_s(x_i - r))\tilde{\mathcal{A}}_i(b)$

Figure 4.3: Player types in BKS.

4.3.1 BKS players

We have already referred informally to the set of players: a single bandwidth seller, and a number of bandwidth buyers. However, a more formal description is in order. Let us define the seller as $s = (k, r)$. The seller has a *reserve price* r, expressed in the same units as bids. Formally, r is fixed over the course of m epochs, but in practice, the seller must be able to occasionally change her reserve price. Furthermore, r is drawn from the finite set R. Section 4.4.2 explains why sellers must have a reserve price chosen from a (small) set of alternatives. The parameter k indicates the seller's buffer capacity. This determines how many packets the seller can store simultaneously before it must start dropping packets. If the seller is forced to drop packets, we say it is *congested*.

Let there be n buyers in the game. These buyers belong to the set $N = \{1, 2, \ldots, n\}$. We use the symbol i to refer to a specific buyer. Each buyer has several associated values. Buyer i has a *type* v_i. This type encodes the value the buyer receives per packet of data. It is private to buyer i and independent of the v_j for all $j \neq i$. While we do not explicitly model buyers' demand for bandwidth, we will at times find it useful to think of each buyer as having a maximum demand for bandwidth within each epoch, to which we refer to as d_i.

4.3.2 A single BKS epoch

As described above, an *epoch* is the set of steps that are repeated over and over in BKS. In each epoch, the seller first solicits bids from its buyers. Then for a period of time we call the *epoch length*, the seller allocates bandwidth to buyers according to their bids. Buyers are charged payments as a function of their bids and their bandwidth allocations. The next epoch begins when the seller solicits new bids from the buyers. We assume that sellers solicit and collect bids from buyers instantly. In practice, the end of epoch t would most likely overlap with the bidding for epoch $t+1$. In general, we will denote a particular epoch with the symbol t.

The description above elides over the differences in the way buyers and sellers participate in each epoch. In Sections 4.3.3 and 4.3.4 we will see explore the differences between the buyer and seller "views" of BKS:

- Buyers' payments are not equal to seller revenues.
- Buyers bid as if they are participating in a Babaioff et al. style random, truthful auction.
- Sellers allocate packets as if conducting a first-price auction.

4.3.3 Buyer view of BKS

The buyer view of BKS is designed to incentivize truthful bidding, as we show in Section 5.3. This incentive comes from assigning payment to buyers according to Equation 3.7. Each epoch of BKS can be seen as an implementation of Babaioff et al.'s mechanism transformation.

In each epoch's bidding phase, the buyer i reports her bid b_i, (assumed to be in units of currency/packet) to the seller. Having reported her bid, the buyer begins transmitting packets through the seller. This is the allocation phase. The buyer reports the receipt of each packet to the market server. In practice, these receipts should be batched to conserve bandwidth. As mentioned in Section 2.2.1, receipts are cryptographically signed with the buyer's private key. This way, the seller can verify that the receipt is correct, but cannot alter the receipt or forge fake receipts. The buyer knows that if she does not forward her receipts, the seller will terminate her connection. After the epoch length passes, the buyer begins the next epoch by reporting her next bid to the seller.

At the end of the epoch, buyer i will have received an allocation $\tilde{\mathcal{A}}_i$ consisting of a number of packets. From $\tilde{\mathcal{A}}_i$, she can compute her realized value $v_i\tilde{\mathcal{A}}_i$. However, i does not know the exact payment P_i she has been charged, as the payment function $P_i(b)$ is randomized (see Equation 3.7). This charge will be aggregated over many epochs. Buyer i is invoiced after m epochs, and only then does she learn her total payment owed. In practice this means that market server sends each buyer a bill at the end of the month.

We can write the utility u_i received by buyer i in each epoch as

$$u_i(b) = v_i\tilde{\mathcal{A}}_i(b) - P_i(b) \tag{4.1}$$

For a known realized allocation and payment, this u_i becomes

$$u_i = v_i\tilde{\mathcal{A}}_i - P_i \tag{4.2}$$

This is a *quasi-linear utility model*, a common concept in the mechanism design literature [13]. This model carries the implication that buyers have no budget constraints. They will make arbitrarily large payments in exchange for an allocation of packets of equal or greater value.

4.3.4 Seller view of `BKS`

We first consider the procedure observed by the seller in each epoch of `BKS`, and the discuss how the seller's payment is computed as a result of this procedure.

In each epoch, the seller first solicits bids from buyers. Together, these bids constitute the bid vector b. The seller then *resamples* b according to Equation 3.5. This yields the resampled bid vector x.

For the duration of the epoch, the seller routes the buyers' packets. When the number of packets buyers wish to transmit exceeds the seller's routing capacity k, she is forced to *drop* packets. Dropped packets are not transmitted and do not contribute to any buyer's allocation. We refer to the seller's routing behavior as *k-priority allocation*:

Definition 4.3.1. Let seller s maintain a packet queue with a maximum capacity of k packets. Let a packet's *priority* be determined by its owner's resampled bid x_i, such that the greatest x_i translates to the top priority. Then *k-priority allocation* is the process of inserting and removing packets from this queue so that packets with higher priority are whenever possible delivered before those of lower priority.

- If the number of packets in the queue is less than k, then an arriving packet is always inserted into the queue.
- If the seller receives a packet at a time when its queue is full, then the seller must drop a packet. The seller chooses the packet with the lowest priority from the set of size $k + 1$ consisting of those packets in its queue and the arriving packet, and drops it. The remaining k packets occupy the queue.
- When removing a packet from its queue in order to route it, the seller always chooses the highest priority packet currently in its queue.

We say that k-priority allocation induces the $\mathcal{O}_{STRICT}$ *routing prioritization*.

Definition 4.3.2. A *routing prioritization* characterizes the order in which packets are delivered by the seller. Formally, it is a function which computes a permutation of a set of packets. In this paper, we will be concerned with two routing prioritizations:

- $\mathcal{O}_{STRICT}$: packets are routed such that higher priority packets are delivered before lower priority packets.
- $\mathcal{O}_{FIFO}$: packets are routed in the order in which they arrive in the queue.

Remark 4.3.1. Under BKS, the seller achieves $\mathcal{O}_{STRICT}$ with respect to the resampled bids x_i.

The seller routes traffic under the assurance that for each byte routed for buyer i, she will receive payment $x_i - \bar{l_s}(x_i - r)$. The constant $\bar{l_s}$ is the *tax rate*, which is determined by the market server. We describe $\bar{l_s}$ in greater detail in Section 4.3.5.

The seller's utility u_s is given by

$$u_s(b) = \sum_{i=1}^{n}(x_i - \bar{l_s}(x_i - r))\tilde{\mathcal{A}}_i(b) \tag{4.3}$$

According to u_s, the seller does not derive any inherent value from her participation in the mechanism. Instead, all her utility is obtained from the payments earned from buyers.

We can re-write the seller's utility as

$$u_s(b) = \sum_{i=1}^{n} x_i \cdot \tilde{\mathcal{A}}_i(b) - \bar{l_s}\sum_{i=1}^{n}(x_i - r)\tilde{\mathcal{A}}_i(b) \tag{4.4}$$

We will refer to the rightmost sum as the seller's *revenue above the reserve price.*

Definition 4.3.3. The seller's *revenue above the reserve price* is given by the expression

$$\sum_{i=1}^{n}(x_i - r)\tilde{\mathcal{A}}_i(b) \tag{4.5}$$

The tax rate $\bar{l_s}$ is applied to the seller's revenue above the reserve price, rather than her total revenue.

4.3.5 Reconciling payments with the market server

As described in sections 4.3.3 and 4.3.4, buyers' net payments are computed differently than the revenue earned by the seller. It is up the market server to reconcile these two quantities (we will see greater justification for this in our discussion of ALIGN-TRUST).

The sum of buyers' net payment, given by Equation 3.7, must be equal to the seller's revenue. The method by which the market server can balance these two sums is through the tax rate $\bar{l_s}$. This is a fixed portion of the seller's revenue above the reserve price. The tax is collected by the market server and redistributed to buyers in order to pay for buyer rebates.

We first examine the procedure by which the market server collects information on buyers' bids and allocations throughout each epoch. Then, we derive an expression for $\bar{l_s}$.

At the beginning of each epoch, the market server must coordinate with the seller to ensure that the she (the seller) correctly resamples the bid vector b. This is because the market server will credit payment to the seller according to the resampled bid vector. Practically speaking, this coordination could involve exchanging a random seed every few epochs. The market server must also note when buyer i's resampled bid falls below its original bid (that is, $x_i < b_i$) meaning that i is due to receive a rebate according to Equation 3.6.

Throughout the epoch, the market server aggregates the buyers' receipts, which are forwarded to it by the seller. The market server uses its stored copies of buyers' public keys to verify the authenticity of the receipts.

After m epochs, the market server can compute $\bar{l_s}$. Before deriving an expression for $\bar{l_s}$, let us define a few useful quantities. Let the superscripts $t = 1, 2, \ldots, m$ denote the epoch number with which a value is associated. For example, $\tilde{\mathcal{A}}_i^t$ is buyer i's allocation in epoch t. An overbar above a symbol signifies that it represents a value which is computed across m epochs, rather than on an epoch-by-epoch basis.

- $\bar{D}$ is the total value of all rebates owed to buyers:

$$\bar{D} = \sum_{t=1}^{m} \sum_{i=1}^{n} R_i^t \tag{4.6}$$

- $\bar{K}$ is the total difference between the per-byte price paid by buyers (before rebate), and the per-byte price received by sellers. That is,

$$\bar{K} = \sum_{t=1}^{m} \sum_{i=1}^{n} \mathcal{A}_i^t (b_i^t - x_i^t) \tag{4.7}$$

- $\bar{V}$ is the total payment to sellers above the reserve price:

$$\bar{V} = \sum_{t=1}^{m} \sum_{i=1}^{n} \mathcal{A}_i^t (x_i^t - r) \tag{4.8}$$

We can compute $\bar{l_s}$ as

$$\bar{l_s} = \frac{\bar{D} - \bar{K}}{\bar{V}} \tag{4.9}$$

In other words, $\bar{D} - \bar{K}$ is the amount of money that needs to be paid to buyers, and $\bar{V}$ is the pool of money from which this payment will be drawn.

4.4 The ALIGN-TRUST subprocedure

Why go to the extra trouble in Sections 4.3.4 and 4.3.5 of instituting an indirect payment system which places a tax on sellers? This exercise in algebra allows the market server to pay buyer i her rebates without collecting them directly from s, buyer i's seller. As we will see in the Chapter 5, we use this indirection to assure RABID's faithfulness with respect to sellers.

4.4.1 Computing tax rates

The purpose of ALIGN-TRUST is to compute a value of $\bar{l_s}$ for each seller s in such a way that $\bar{l_s}$ is minimized by faithful routing according to $\mathcal{O}_{STRICT}$. This subprocedure is performed by the market server. In previous subsections, we discussed the interaction between the market server and a single seller and her corresponding set of buyers. In reality, many independent sellers simultaneously mediate their interactions with their buyers through the market server. Let S be the set of all sellers served by the market server, and let $s \in S$ refer a specific seller. Furthermore, let us randomly partition S into S_1 and S_2 such that $S_1 \cup S_2 = S$, $S_1 \cap S_2 = \emptyset$, and $|S_1| \approx |S_2|$. In Section 4.2.1, we introduced a values $\bar{D}$, $\bar{K}$, and $\bar{V}$ (Equations 4.6 to 4.8). Now let us define similar values over all sellers in S_k for $k \in \{1, 2\}$:

$$\bar{D}_k = \sum_{s \in S_k} \bar{D}_s \tag{4.10}$$

$$\bar{K}_k = \sum_{s \in S_k} \bar{K}_s \tag{4.11}$$

$$\bar{V}_k = \sum_{s \in S_k} \bar{V}_s \tag{4.12}$$

Let $\bar{l_s}^{(k)}$ be the tax rate for all sellers in set S_k. We can now formally describe ALIGN-TRUST.

Definition 4.4.1. The ALIGN-TRUST procedure simultaneously determines the tax rates $\bar{l_s}^{(k)}$ for $k \in \{1, 2\}$ corresponding to seller pools S_1 and S_2. It computes $\bar{D}_k$, $\bar{K}_k$ and $\bar{V}_k$, and then sets taxes rates according to

$$\bar{l_s}^{(k)} = \frac{(\bar{D}_{(-k)}) - (\bar{K}_{(-k)})}{\bar{V}_k} \tag{4.13}$$

4.4.2 Pooling sellers by reserve price

We have omitted one last detail. Sellers will need to choose different reserve prices – sellers who routinely hang around in large, crowded rooms full of lawyers and bankers should set a higher reserve price than those whose primary clientele consists of undergraduate students.

Throughout our description of `ALIGN-TRUST`, we implicitly assumed that r was identical for all $s \in S$. However, not all sellers will set the same reserve price. In order to accomodate the heterogeneity of reserve prices, the market server must actually perform `ALIGN-TRUST` not just once, but instead one time for each of the reserve prices in R. We can think of the set S in align trust as the set of all sellers for a given reserve price $r \in R$. The need to pool sellers by reserve price dictates that we select a small, discrete set of legal reserve prices.

Chapter 5

Theoretical analysis of RABID

Having described RABID in detail, we are now ready to consider how it achieves the desirable economic properties introduced in Section 3.1. In the course of this analysis, we will introduce two other mechanisms: VMM and FIXED. This pair of alternatives will offer illustrative comparisons to RABID. First, we will introduce FIXED and VMM. Then, we will show how RABID achieves constrained efficiency, universal ex-post individual rationality for buyers, individual rationality in expectation for sellers, strong budget balance in expectation, seller faithfulness, and buyer truthfulness in expectation.

5.1 Alternative mechanisms

In order to establish some context for RABID, we consider two other mechanisms which do not achieve all the properties of RABID. FIXED is a baseline mechanism in which packets are allocated at a fixed price on a first-come, first-serve basis. VMM is named after Hal Varian and Jeffrey MacKie-Mason, whose work [15] this mechanism derives from. It conducts a packet-level VCG auction for each packet that enters the seller's routing queue.

5.1.1 FIXED

In this mechanism, we do not consider seller incentives. The price per packet is fixed at p. Buyers must pay exactly p to send or receive a packet. Buyer utility is given by

$$u_i^{\texttt{FIXED}} = (v_i - p)\mathcal{A}^{\texttt{FIXED}} \tag{5.1}$$

Routing of packets is performed on a first-in, first-out basis. We call the induced prioritization of this allocation rule $\mathcal{O}_{FIXED}$.

`FIXED` is truthful (see Definition 3.1.6) and universally ex-post individually rational (Definition 3.1.2) for buyers, under the reasonable assumption that buyers can control their demand. It is weakly budget balanced (Definition 3.1.4). However, it is critically inefficient.

5.1.2 `VMM`

In Sections 1.2 and 2.2.3, we allude to the failure of VCG mechanisms to be truthful in repeated settings. `VMM` is one such mechanism. It is similar to the `BKS` procedure described in Section 4.3. In each epoch, buyers report their bids to the seller, who then allocates bandwidth to the buyers over the course of the epoch. As in `BKS`, the seller is defined as $s = (k, r)$, where k is the seller's buffer capacity and r its reserve price. Allocation is performed according to k-priority allocation. However, `VMM` does not resample bids, and therefore performs k-priority allocation with respect to the original bid vector b.

Buyers are charged according to the VCG payments. Varian and MacKie-Mason argue in [14] that the VCG price for any packet j is the bid of the highest-priority packet j' dropped while j was in the queue (where j' may be the packet displaced by j on its entrance into the queue).

Definition 5.1.1. Let $o, o', \ldots$ be a series of times at which packets may arrive at the `VMM` seller and potentially be inserted into the seller's queue. Let $O(o_1, o_2)$ be the set of all packets dropped by the seller between times o_1 and o_2. Then the *`VMM` price* of a packet p arriving at the seller at time o_1 and routed at time o_2 is given by the maximum of the bids associated with each packet in the set $\{j' | j' \in O(o_1, o_2) \text{ and } owner(j') \neq owner(j)\}$.

`VMM` is universally ex-post individually rational and weakly budget balanced. Unlike `FIXED`, It is constrained-efficient (Definition 3.1.1) in the case that buyers bid truthfully. As mentioned above, `VMM` is not truthful for buyers. Despite `VMM`'s use of the Vickrey-Clarke-Groves mechanism for allocating and pricing packets, its repeated application of VCG disincentivizes buyers' truthful bidding.

5.2 Constrained efficiency

`RABID` produces an approximately constrained-efficient allocation. Because `RABID`'s subprocedure `BKS` allocates bandwidth according to the the resampled bids x_i rather than the original bids b_i, inefficient allocations sometimes occur. Babaioff, Kleinberg, and Slivkins provide an excellent starting pointing for analyzing `RABID`'s efficiency. In Section 3.5 of their 2010 paper [2], the authors place bounds on their transformation's welfare. Since

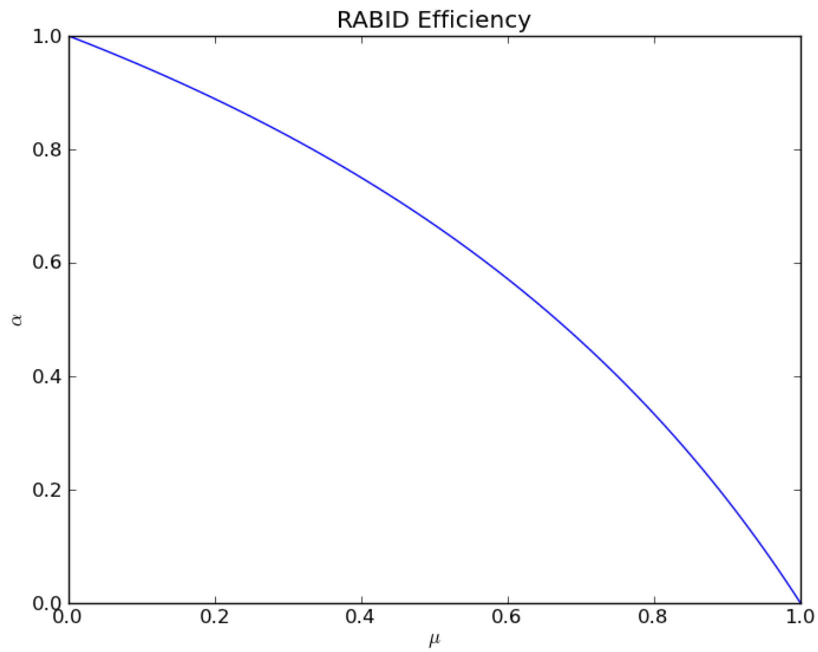

Figure 5.1: Increasing μ lowers α, the expected value of the ratio of `RABID`'s efficiency to the optimal efficiency, assuming a reserve price of 0.

the transformation lies at the heart of `RABID`, much of their findings are applicable in this situation.

The mechanism parameter μ gives the probability that $x_i < b_i$ for some i in some epoch. Babaioff et al. point out that by the union bound, the probability that $\exists x_i < b_i$ is at most $n\mu$. Therefore, in any particular epoch, the allocation will be fully efficient (under both Babaioff et al.'s transformation and `RABID`) with probability at least $1 - n\mu$.

We are also interested in the ratio of `RABID`'s average efficiency to the efficiency of the optimal allocation. Babaioff et al. show that for reserve prices $r = 0$, the expected value of this ratio is given by $\alpha = 1 - \mu/(2 - \mu)$.

5.3 Buyer truthfulness

We now consider the truthfulness (in expectation) of `RABID` for buyers.

Theorem 1. *`RABID` is truthful for buyers in expectation.*

Proof of Theorem 1 follows from the application of Babaioff et al.'s generic transformation to k-priority allocation (Definition 4.3.1), which yields the buyer payments. To show this, we must prove that k-priority allocation w.r.t. buyer i's original bid b_i constitutes a monotone allocation rule.

Lemma 2. *k-priority allocation w.r.t. b_i is a monotone allocation rule.*

Proof. We must show that for fixed b_{-i} and for all b_i' such that $b_i \geq b_i'$, k-priority allocation yields $\mathcal{A}_i(b_i, b_{-i}) \geq \mathcal{A}_i(b_i', b_{-i})$. Consider that all packets allocated at bid b_i' will also be delivered for the weakly greater bid b_i. Furthermore, there may exist packets which would be dropped at bid b_i' but delivered at b_i. Thus, $A_i(b_i, b_{-i}) \geq A_i(b_i', b_{-i})$, completing the proof.

In comparison to `RABID`, we know that `VMM` is not truthful for buyers with over the course of an epoch. This is due to `VMM`'s use of VCG pricing, which is known not to be truthful in repeated settings [5]. The repetition in `VMM` stems from the large number of packets that are routed in each epoch. The routing of each packet constitutes a single VCG auction. As a result, buyers will have an incentive to shave their bids.

`FIXED`, on the other hand, achieves complete buyer truthfulness. This follows trivially from the payment rule. Let us interpret the price p as an agent i's bid, such that $b_i = p$ and

$$P_i^{\texttt{FIXED}} = p\mathcal{A}_i = b_i\mathcal{A}_i - \int_{-\infty}^{b_i} \mathcal{A}_i(u, b_{-i})du \tag{5.2}$$

Since no allocation is possible at any bid less than p, we have that

$$\int_{-\infty}^{b_i} \mathcal{A}_i(u, b_{-i})du = 0 \tag{5.3}$$

Hence, by the Myseron characterization of truthful payment, `FIXED` is truthful under all circumstances.

5.4 Universal ex-post individual rationality for buyers

Individual rationality for buyers in `RABID` follows from Babaioff et al.'s generic transformation. Buyers never need bid (or pay) more than their value per bytes.

This is also the case in `VMM`, which leads to individual rationality for buyers in this mechanism as well.

In `FIXED`, buyer i only demands bandwidth when $v_i > p$, thus yielding individual rationality for buyers.

5.5 Individual rationality in expectation for sellers

ALIGN-TRUST introduces some complexities which seem to threaten the budget balance of RABID. Luckily, we can show that in the limit where the number of sellers grows to infinity, our mechanism is indeed individually rational for sellers.

Theorem 3. *Let S be the set of sellers over which revenues are pooled and taxed by ALIGN-TRUST. Then, for all $s \in S$,*

$$\lim_{|S| \to \infty} u_s \geq 0 \tag{5.4}$$

Proof. Remember that ALIGN-TRUST randomly partitions S into sets S_1 and S_2. This trick is due to Goldberg and Hartline's work on competitive auctions [6]. For each set, we compute the sum of the rebates $\bar{D}_k$ owed to buyers associated with sellers in set S_k, as well as $\bar{K}_k$, the total difference between buyers' and sellers' per-packet payment. We also computed $\bar{V}_k$, the total pre-tax revenue of sellers in set S_k. We can think of these values as averages taken over the randomly chosen sets S_1 and S_2. Hence, by the law of large numbers,

$$\lim_{|S| \to \infty} \{\bar{D}_1, \bar{K}_1, \bar{V}_1\} = \lim_{|S| \to \infty} \{\bar{D}_2, \bar{K}_2, \bar{V}_2\} \tag{5.5}$$

$$\lim_{|S| \to \infty} \bar{l}_s^1 = \lim_{|S| \to \infty} \bar{l}_s^2 \tag{5.6}$$

In this case, the tax rate is the same across the entire set of sellers. But since we know that paying the tax rate determined for paying off one's own buyers is guaranteed to be less than or equal to one. As a result, the expectation of seller revenue, and therefore, utility, is non-negative.

Lemma 4. *Sellers can expect per-packet revenues greater than or equal to the reserve price r:*

$$\mathbf{E}\left[\frac{u_s}{\sum_{t=1}^{m} \sum_{i=1}^{n} \tilde{\mathcal{A}}_i^t}\right] \geq r \tag{5.7}$$

Proof. By Theorem 3, we have that $\mathbf{E}[\bar{l_s}^{(k)}] \leq 1$. Substituting $\bar{l_s}^{(k)} = 1$ into Equation 4.3, we obtain

$$u_s(b) = \sum_{i=1}^{n} (x_i - (1)(x_i - r))\tilde{\mathcal{A}}_i(b) \tag{5.8}$$

$$u_s(b) = r \cdot \sum_{i=1}^{n} \tilde{\mathcal{A}}_i(b) \tag{5.9}$$

By summing over m epochs and re-arranging, we obtain Equation 5.7, completing the proof.

5.6 Seller faithfulness

We will now examine the seller faithfulness of `RABID` (see Definition 3.1.5). A naive application of Definition 3.3.2 without `ALIGN-TRUST`, while intended to achieve priority allocation, incentivizes the seller to violate the intended prioritization, even in expectation. If the seller is aware that buyer i's resampled bid x_i has been resampled below the original bid b_i, then she knows i will receive a rebate in the current epoch. Since $R_i = b_i \cdot \tilde{\mathcal{A}}/\mu$, the seller will "cut off" buyer i until the end of the epoch.

The seller s must be compelled to adhere to the $\mathcal{O}_{STRICT}$ routing prioritization. The seller payment rule should ensure the seller receives the greatest payment per packet for packets belonging to the highest bidder, the second greatest payment for those belonging to the second highest, and so on.

Theorem 5. *`RABID` is faithful for sellers.*

Proof. In order to prove Theorem 5, we must show that sellers maximize their revenues by implementing $\mathcal{O}_{STRICT}$ with respect to x.

The price per packet paid to sellers is equal to the resampled bid x_i, minus a tax determined independently of the seller's allocation. Routing the most valuable packets first pays no less than routing less valuable packets first. Finally, a seller cannot decrease her tax rate $\bar{l_s}^{(k)}$ by decreasing her revenue. Therefore, sellers receive maximum utility by routing according to $\mathcal{O}_{STRICT}$ with respect to x.

This property holds for `VMM` as well. Since packets with higher bids are always routed first, and `VMM` payment is a monotone function of bid, sellers participating in this mechanism will have no incentive to distort the routing prioritization.

Trivially, `FIXED` is faithful for sellers since all packets yield exactly the same utility.

5.7 Strong budget balance in expectation

Strong budget balance in expectation is a system-wide property of `RABID`. Intuitively, it is strongly budget balanced since no cash need enter or leave the system so long as all seller tax rates $\bar{l_s}^{(k)} \leq 1$.

5.8 Fairness and RABID's parameter μ

The parameter μ dictates the probability any particular agent's bid is resampled in some epoch. As discussed in Section 5.2, the frequency with which bids are resampled bears on the efficiency of the mechanism. Resampling (usually) reorders the priority on buyers' traffic. In Babaioff et al.'s transformation, the choice of μ represents a tradeoff between efficiency and rebate size. But in `RABID`, tuning μ has a slightly different effect.

As μ approaches zero, individual rebates become less frequent and greater in magnitude. Fixing the time span (number of epochs) over which a seller operates, this translates to increased variance in revenue. Imagine a seller who operates for fifty epochs with $\mu = 0.01$. If the seller must pay even a single rebate, this rebate may be larger than the sum of the seller's revenue. Under `ALIGN-TRUST`, the seller is not responsible for paying this rebate. As a result, the seller can almost certainly expect positive revenue, rather than the grossly negative revenue she would have faced without `ALIGN-TRUST`'s rebate pooling. Through this example we see that μ trades off efficiency for fairness – as μ approaches zero, the mechanism's efficiency increases, but the strength of the redistributive effect increases as well. In Section 6.3.3, we present experimental evidence in support of this claim.

Chapter 6

Experimental results

Until this chapter, this thesis has been concerned with the theoretical properties of `RABID`. But we are also interested in characterizing the mechanism's behavior through simulation. We have constructed a packet-level simulator of `RABID`, and in this chapter we present our simulation results. In Section 6.1, we describe our simulator. Then in Section 6.2, we provide experimental evidence demonstrating `RABID`'s theoretical properties. Finally, in Section 6.3, we explore behaviors of `RABID` revealed by our simulations.

6.1 Simulator overview

Our simulator is designed to study the allocation of bandwidth in a reasonably realistic setting. We seek to model the arrival, queuing, and delivery of packets. By conducting simulations, we seek to explore how `RABID` allocates packets under congestion, measure the revenue it achieves, and observe the variance in the utility received by its participants.

6.1.1 Simulation scope

We have chosen the scope of our simulator to balance simplicity and performance with realism and scale. Our simulator is limited to the `BKS` sub-procedure of `RABID`. This means that we model only a single seller and a small number of buyers to whom she sells bandwidth. Because our simulated seller always routes according to the priority ordering stipulated by `BKS`, we can combine the results of multiple simulations to observe the results of applying `ALIGN-TRUST`, `RABID`'s other sub-procedure.

Each simulation runs for a set number of epochs (usually fifty). Once the simulation is complete, we aggregate data on packets delivered and dropped, payments disbursed, and utility achieved. We then export this data to an analysis suite for visualizing our simulation results. Each epoch has a specified epoch length (usually 300 milliseconds). Because many of the elements of the simulator are randomized, most of our simulated experiments aggregate the results of several simulations.

6.1.2 Simulated epochs

At the simulator's core is its treatment of each single epoch. Each epoch begins at some time t measured after the beginning of the simulation, and ends at $t+l$, where l is the epoch length. At the beginning of the epoch, the seller solicits bids from each buyer. It provides buyers with data about system behavior in previous epochs, including other agents' bids and their allocations. This allows us to implement highly strategic buyer behavior. However, most buyer bidding strategies do not require this information. The buyer strategy we experiment with most commonly is truthful bidding. We implement the buyers' bidding behavior through the combination of a *strategy model* and a *utility model*. The utility model dictates the buyer's value per packet, which can then inform the strategy model's choice of bid.

Once bids have been collected, the seller solicits packets from buyers. We ignore the direction of these packets – there are no separate upload and download queues, only a single unidirectional queue. We also ignore the destination of the packets. The seller will simply attempt to deliver each packet back to the buyer who submitted it. This is a simple way to keep track of which packets are actually routed, and which are dropped. Buyers submit packets to the seller based on a *demand model*. Unless otherwise noted, we implement this demand model as a random packet generator. The time delay between packets is an exponentially-distributed random variable. We control demand through the exponential distribution's rate parameter λ, where a greater λ translates to a shorter delay between packet transmissions.

The seller asks buyers to submit all packets they wish to transmit before the next *packet collection delay*. At each delay, typically every 100 milliseconds, the seller removes a number of packets from its queue and delivers them to buyers. We call the number of packets delivered the *packets per delay*. Under the aforementioned buyer demand model with an exponentially distributed time delay between packets, the number of packets arriving in each packet collection delay follows a Poisson distribution.

At the end of each epoch, the seller tallies the packets delivered and dropped, and computes the payment from each buyer. We store a set of statistics for each epoch, including the number of packets delivered, the number of packets dropped, the payment made in each period, and the value received by each agent.

6.1.3 Simulating `ALIGN-TRUST`

As mentioned in Section 6.1.1, our simulator does not perform the `ALIGN-TRUST` procedure. However, it is straightforward to emulate `ALIGN-TRUST` in the analysis of simulation data. Because simulated sellers always route packets according to design, we can skip the random assignment of sellers to different pools, and simply compute the tax rate over all sellers. The resulting seller revenues are essentially equivalent to those obtained through `ALIGN-TRUST`.

6.1.4 Default simulation parameters

Our experiments typically vary a handful of parameters. However, the majority of parameters remain unchanged. In Sections 6.2 and 6.3, we describe our experiments in terms of modified parameters. Understanding our experiments therefore requires an understanding of our default parameters. These are summarized in Figure 6.1.4.

System-wide parameters

Parameter name	**Default value**
μ	0.1
epoch length	300ms
packet delivery delay	150ms
max. packets delivered per delay	300
seller buffer capacity	20
reserve price	0.75

Buyer parameters

	v_i	b_i	λ
Buyer 0	1	1	0.065
Buyer 1	2	2	0.065
Buyer 2	3	3	0.065
Buyer 3	4	4	0.065

Figure 6.1: Summary of our default simulation parameters.

Our choice of default parameters leads to several default behaviors. Each buyer generates an average of 20 packets per epoch. The seller can route 40 packets per epoch, and therefore can fully serve only two buyers at a time. In our default experimental setup, participants experience network congestion.

6.1.5 Implementation details

Our simulator is implemented in Haskell. The choice of language has had an important impact on the development of our simulator. Haskell's extraordinary type system has been a huge asset in ensuring correctness and finding bugs at compile time. The excellent performance of native code generated by GHC enables us to run experiments quickly. Each individual simulation completes in hundredths of a second. Additionally, Haskell's dense standard library and wonderful package manager have proven to be tremendous time savers.

6.2 Simulation results concerning theoretical properties

In this section, we present our simulation results in support of our theoretical claims concerning `RABID`. We will focus on the properties most evident in simulation, namely, efficiency, truthfulness, and individual rationality.

6.2.1 Efficiency

`RABID` is designed to produce efficiency gains over fixed-price mechanisms by allocating bandwidth to buyers with the greatest value for data. In Figure 6.2.1, we demonstrate `RABID`'s efficiency gains compared to `FIXED` as we increase the seller's buffer size. When the seller's buffer is small, the seller is forced to drop packets. Under `FIXED`, the seller drops packets without regard for their value. The packets dropped are essentially chosen at random, since packets arrive at random times. However, `RABID` and `VMM` both keep higher-value packets in their queues and drop those of lower value.

Interestingly, `VMM` achieves higher efficiency than `RABID` in the data shown in Figure 6.2.1. This is due to a loss of efficiency from the resampling procedures at the heart of `BKS`. However, we see that the difference in efficiency is small, substantiating `RABID`'s approximate efficiency.

Yet `VMM`'s efficiency in this context is somewhat artificial, since in this experiment buyers bid truthfully. As we will see in Section 6.2.2, buyers under `VMM` generally receive a boost in utility from shaving their bids by the right amount. Figure 6.2.1 shows that `VMM` with truthful bidding achieves lower buyer utility than `RABID`.

6.2.2 Buyer truthfulness in expectation

Figure 6.2.2 compares the utility of a strategic buyer under `VMM` and `RABID`. In this experiment the strategic buyer, Buyer 1, has a value of 5 per packet. She competes for limited bandwidth against Buyer 0, who has value 4 per packet, but demands slightly more data than Buyer 1. Buyer 0 always bids truthfully, while Buyer 1 bids strategically. Buyer 1's bid is given on the x-axis in Figure 6.2.2's. When Buyer 1 bids below Buyer 0, she receives greater utility than she would truthfully bidding her value per packet of 5. Focusing on the

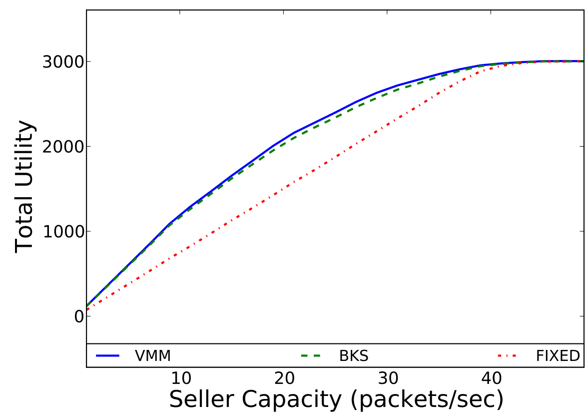

Figure 6.2: Under congestion, `RABID` and `VMM` both allocate bandwidth more efficiently than `FIXED`.

efficiency achieved by the mechanism, we see that Buyer 1's strategic behavior has a slight negative effect on the mechanism's utility. When Buyer 1 raises her bid above Buyer 0's, the mechanism's total utility (efficiency) increases.

6.2.3 Revenue and weak budget balance

It is important that our simulation results bear-out our claim that `RABID` is individually rational for sellers in expectation. We would like to show not only that sellers can expect not to lose money, but that under reasonable circumstances, they should expect strongly positive revenues. Figure 6.2.3 demonstrates that even in the absence of a reserve price, sellers can achieve strong revenue under `RABID`. To generate this figure, we run numerous simulations, each with randomly chosen parameters. We vary the number of buyers uniformly at random between one and ten. Buyers' values-per-packet are sampled uniformly at random on the interval $[0, 5]$. We use the resulting data to emulate `ALIGN-TRUST`, as described in Section 6.1.3.

The revenue distribution is skewed heavily to the right. Not only are all sellers' revenues positive, but in fact the majority of sellers enjoy healthy revenue.

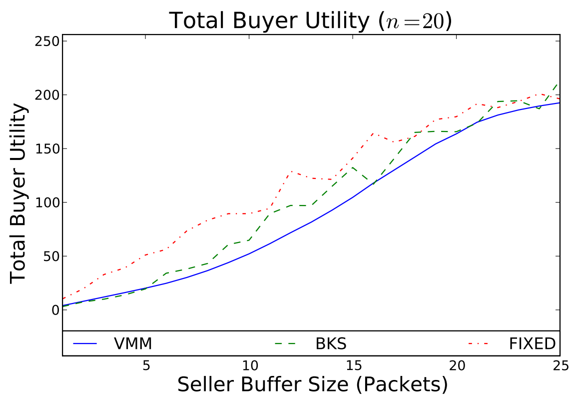

Figure 6.3: VMM achieves lower buyer utility under truthful bidding than BKS, as truthful bidding in VMM is not a dominant strategy. As a result, truthfully bidding buyers in VMM overpay for their bandwidth.

6.3 Simulation results characterizing the behavior of RABID

6.3.1 ALIGN-TRUST decreases revenue variance

ALIGN-TRUST is a necessary part of RABID– without it, seller faithfulness would not be a dominant strategy. But ALIGN-TRUST confers additional benefits on sellers. It vastly reduces the variance in revenue over BKS alone (assuming that buyers do not behave strategically in this instance of BKS). The pooling performed by ALIGN-TRUST averages out much of the variance in payments that arises from the large, infrequent random rebates generated by BKS. On the x-axis of Figure 6.3.1 are revenues that sellers would achieve without ALIGN-TRUST, assuming faithful behavior. The y-axis shows revenue under ALIGN-TRUST. Each point represents a single seller. The vertical compression which is evident in this data set demonstrates the reduction in variance of seller revenues. Note that the range of the y-axis in this figure is only one-fourth that of the x-axis. We also observe that while sellers appearing left of $x = 0$ would have suffered negative revenues alone, ALIGN-TRUST ensures that all sellers achieve non-negative revenue.

6.3.2 Seller revenue

Our simulation results provide some characterization of RABID's revenue generating behavior. In Figure 6.3.2, we vary the seller's buffer capacity between 1 and 25. We do not apply the ALIGN-TRUST transformation in post processing, so that the figure shows the results a well-behaved seller could expect without rebate pooling and at reserve price $r = 0$. We see

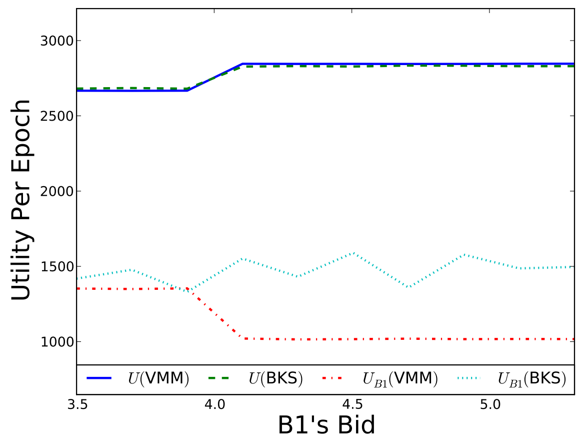

Figure 6.4: Buyer 0 has value 4; Buyer 1 has value 5. We see that Buyer 1 maximizes her utility under `VMM` by under-bidding. Unfortunately, this causes a reduction in the `VMM`'s efficiency. Under `BKS`, Buyer 1 maximizes her utility by bidding truthfully.

that revenue is greatest when capacity is high enough to let a reasonable number of packets reach their destination, but also low enough that bandwidth remains a scarce, expensive resource.

Figure 6.3.2 shows results from an experiment similar to experiment reported on in Figure 6.3.2, with the difference that Figure 6.3.2 modifies the seller's reserve price rather than her buffer capacity. Because of the congestion mechanism participants under our default parameters (see Figure 6.1.4), bandwidth is already a scarce and expensive resource. Figure 6.3.2 shows that under these conditions, it is expensive enough that no fixed-price auction can achieve greater revenue.

6.3.3 Tuning μ

In Section 5.8, we discussed how the mechanism designer's choice of μ represents a trade-off between efficiency and fairness. Figure 6.3.3 explores this effect. To generate this figure, we ran several hundred simulated experiments. As in Figure 6.3.1, each point on the plot represents the outcome of a single experiment. Simulation parameters are randomly selected similarly to the manner in which they are chosen to generate Figure 6.2.3. For these experiments, the number of buyers is chosen uniformly at random between 2 and 6 (inclusive). Buyer values per packets are chosen uniformly at random from $[0, 5]$. Buyer rate parameters (λ) are fixed at 0.065. All other parameters correspond to the defaults. We then generate two groups of experiments. The first is run at $\mu = 0.1$, while the second is

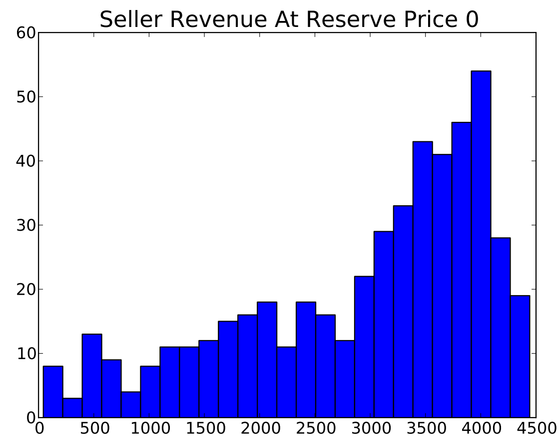

Figure 6.5: Under `ALIGN-TRUST`, no seller loses money.

run at $\mu = 0.9$.

In Figure 6.3.3, we plot the revenue each seller achieves under `RABID` on the y-axis, and the revenue she would achieve without pooling on the x-axis. The distribution of revenue is vertically compressed due to the redistribution engendered by `ALIGN-TRUST`. If there were no redistribution, then every point would fall on the line $x = y$. This allows us to visualize the magnitude of the redistribution by measuring the slope of the data generated by each group of experiments. We see that the slope of of the experiments for $\mu = 0.9$ is greater than that for $\mu = 0.1$, but is still less than 1. That is, a greater μ leads to less utility redistribution but seems unlikely to eliminate this redistribution. It should be noted that $\mu = 0.9$ is an extreme parameter choice – this equates to almost constant resampling, which leads to very poor efficiency.

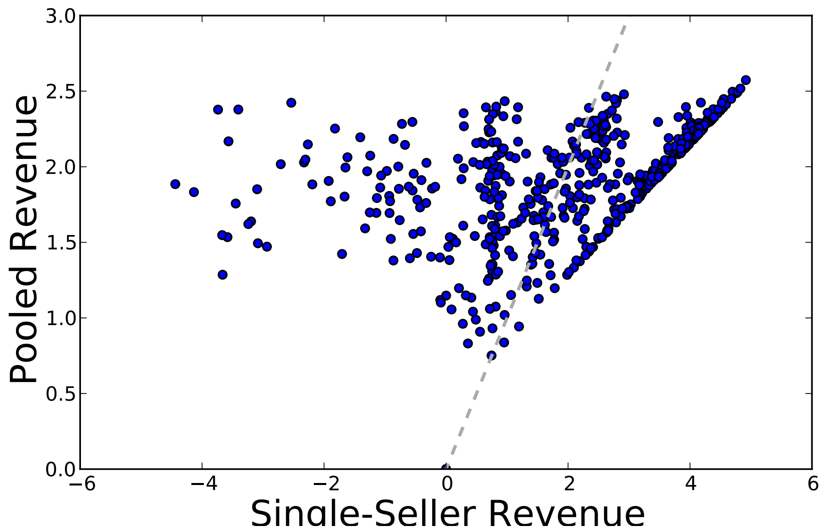

Figure 6.6: `ALIGN-TRUST` reduces the variance in seller revenue. Seller revenues with (y-axis) and without (x-axis) `ALIGN-TRUST` are plotted against each other. The vertical compression evident in this figure demonstrates the reduction in revenue variance.

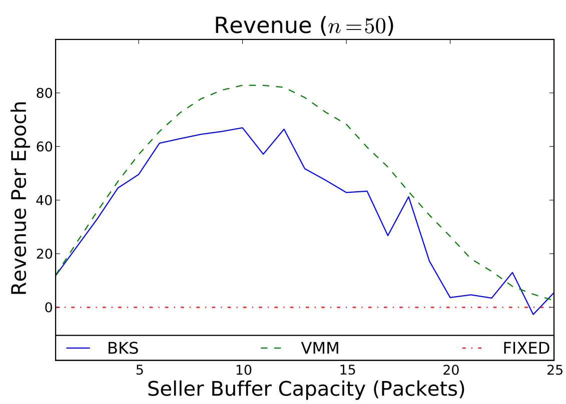

Figure 6.7: For a reserve price of zero, the fixed price mechanism achieves no revenue, but both `VMM` and `RABID` achieve strongly positive revenue.

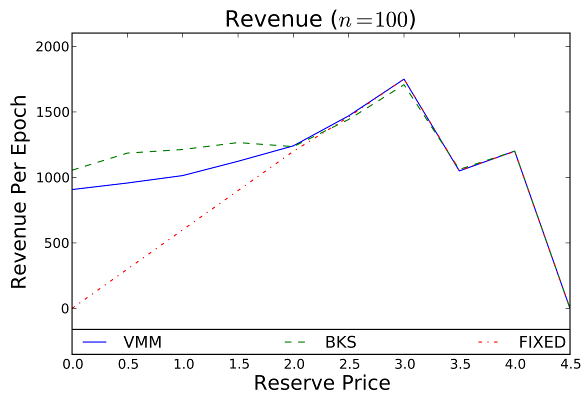

Figure 6.8: Manipulating the reserve price yields no greater benefit to `FIXED` than to `RABID` or `VMM`.

Figure 6.9: As μ increases, `RABID`'s redistributive effect decreases.

Chapter 7

Discussion and Conclusion

In this chapter, we will examine issues which our work does not directly address or which remain unresolved. Among these issues are fairness and collusion. We identify two aspects of `RABID` which raise questions as to whether the allocations and payments computed by the mechanism are fair. Next, we investigate several collusion schemes with potentially dire consequences for seller individual rationality and system budget balance. Resolving questions of fairness and preventing collusion provide fertile grounds for future work.

We end the chapter with some concluding thoughts on this thesis.

7.1 Fairness

The concept of fairness in mechanism design is often fluid, having no single definition. What would characterize a fair bandwidth trading mechanism? Are agents participating in `RABID` treated fairly? Fairness may not be a binary property. Instead, it may be possible to trade fairness for increased efficiency or revenue.

We identify two aspects of `RABID` which raise questions about fairness in the context of bandwidth trading. The first is the redistributional aspect of `ALIGN-TRUST`. By taxing sellers a portion of their revenue, `ALIGN-TRUST` redistributes wealth. Is it fair to sellers to take more from some than from others?

Buyers have fairness concerns as well, centering around the effects of the bid resampling needed to achieve truthfulness. Imagine a buyer with a high value for data and a great demand for bandwidth. If this buyer's bid is resampled below those of a number of other buyers, she might appear to lose her connectivity for the length of an epoch.

We will examine the consequences of these fairness issues, and explore possible tradeoffs in both circumstances between fairness and other properties.

7.1.1 Redistribution

The `ALIGN-TRUST` procedure ensures the faithfulness of `RABID` with respect to sellers. However, its redistributive natures may require participants to accept a degree of unfairness, as a share of their wealth will be taken from them.

We suggest a few ways to redress this unfairness, potentially at the expense of other auction properties.

- Institute a regressive tax rather than a flat tax.
- Inject cash into the system. The extra cash could be used to pay buyers' rebates without taking as much money from sellers. This would necessarily remove all budget balance from the mechanism.
- Do not run `ALIGN-TRUST`, instead let sellers pay buyers' rebates. Removing `ALIGN-TRUST` would dismantle the incentives required for seller faithfulness. However, this may make sense if the mechanism designer believes the technical challenge of implementing manipulative seller behavior would be greater than the benefit this would incur for the manipulator.

7.1.2 Interrupted connectivity

As an unfortunate consequence of the bid resampling we first described in Section 3.3, buyers with a high value for data may be "cut off" from time to time as their bids are resampled below others'. Imagine a consultant making an international voice-over-IP call to a client. The consultant would be very upset if her call were dropped due to having her bid resampled below four other buyers' bids. We must ask if it is fair to let buyers willing to pay dearly for every packet lose their connections.

The ideal solution to this problem is to a design a new self-resampling procedure (see Section 3.3.4) which guarantees that no buyer entirely loses connectivity. However, this seems like a difficult task.

Tuning mechanism parameters offers the simplest hope of addressing this issue. By decreasing both the resampling probability μ and the epoch length, the mechanism designer could reduce the frequency with which buyers are cut off, and decrease the length of time during which they lose connectivity. This would, however, come at the cost of decreased fairness to seller.

Finally, it might be possible to make changes to `RABID`'s allocation rule in order to trade efficiency for fairness. By always routing a certain amount of traffic for each buyer in each epoch, a modified mechanism could avoid cutting off any of its buyers.

7.2 Collusion

We first alluded to the issue of collusion while discussing prior work in Section 1.2, and again in Section 2.2.4 while describing the challenges addressed by `RABID`. In this section, we briefly sketch two collusion schemes and then discuss their consequences.

7.2.1 Buyer-buyer collusion

In this scheme, a set of trusted buyers collude in order to minimize their net payment for bandwidth. The buyers communicate their true values to each other. They then submit reduced bids in such a way as to preserve their true priority order. If all buyers associated with some seller participate in the collusion scheme, buyer i can bid $r + \epsilon_i$ such that $\epsilon_i \ll r$ and pay little more than the reserve price.

Buyers face two obstacles to achieving this type of collusion. First, buyers need to trust each other. Second, they must find a way to communicate their true values to each other. Unfortunately, buyers will always be physically near to each other in order to connect to the seller's wifi adapter, which makes communicating and overcoming trust barriers easy.

Should mechanism designers attempt to create elaborate counter measures for this kind of collusion? By performing statistical inference on buyer bidding behavior or observing communication between buyers, it might be possible to detect and prevent buyer-buyer collusion.

These efforts may not be cost-effective. The main effect of buyer-buyer collusion is to reduce seller revenue. However, it leaves all other properties essentially unchanged. In the worst case scenario, sellers are essentially forced to set a fixed price, and buyers still achieve constrained-efficient outcomes. It may be best to ignore this type of collusion.

7.2.2 Buyer-seller collusion

Buyer-seller collusion is potentially much more harmful to RABID's desirable properties. In buyer-seller collusion, buyers and seller cooperate in maximizing buyers' rebates. The seller notifies a buyer each time she, the buyer, can expect to receive a rebate. In response, the buyer rapidly generates receipts for non-existent packets. The seller forwards these receipts to the market server, delivering a tremendous rebate to the buyer. Since the cost of these rebates is not paid by the seller, she has no disincentive to participate in this scheme. Furthermore, the buyer can give a share of her proceeds to the seller as compensation.

While we refer to this procedure as "collusion," it could in reality be the product of a single agent operating multiple devices, for example, a smartphone and a laptop. In essence, agents can effectively steal from the system. The result is potentially devastating. Buyer-seller collusion might void sellers' expectations of individual rationality, break budget balance, and seller disincentivize faithfulness.

One countermeasure against this attack is to conceal knowledge of which agents will receive rebates in which periods. For example, all bid resampling could be performed by the market server, rather by sellers, who would instead only learn the prioritization order over buyers in each epoch. This would prevent the single-user smartphone-and-laptop attack described above. However, it is unlikely to be completely effective at preventing all buyer-seller collusion attacks. Mechanism designers should expect to need to deploy significant countermeasures against this class of attacks.

7.3 Conclusion

In this thesis, we have presented RABID, a mechanism which facilitates bandwidth trading between untrusting internet device users. RABID addresses a number of challenges inherent to the bandwidth trading context. Among these challenges we count the inability of buyers and sellers to hold each other accountable for delivering promised bandwidth, and the impossibility of computing counterfactual bandwidth allocations. Despite these challenges, RABID achieves a number of desirable economic properties. We prove that RABID is approximately constrained-efficient, universally ex-post individually rational for buyers, individually rational in expectation for sellers, strongly budget balanced in expectation, and truthful in expectation for buyers.

In order to achieve these properties, RABID leverages work by Babaioff, Kleinberg, and

Slivkins to allocate bandwidth on a packet-by-packet basis according to a routing prioritization dictated by buyers' bids. Truthful payments are computed without the need for computing counterfactual allocations.

We provide simulation data in support of our theoretical results. Our simulations compare the performance of `RABID` two alternative mechanisms, `VMM` and `RABID`, and show that `RABID` generally outperforms these alternatives in terms of generating simultaneously truthful and efficient outcomes. Through simulation, we provide additional characterizations of `RABID`'s behavior, illustrating trade-offs between efficiency and fairness.

Bibliography

[1] Wi-fi overload at high-tech meetings. http://www.nytimes.com/2010/12/29/technology/29wifi.html.

[2] M Babaioff, RD Kleinberg, and A Slivkins. Truthful mechanisms with implicit payment computation. *Proceedings of the 11th ACM conference on Electronic commerce*, pages 43–52, 2010.

[3] Y Bachrach. Honor among thieves: collusion in multi-unit auctions. *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 617–624, 2010.

[4] G Biczók, L Toka, A Vidacs, and TA Trinh. On incentives in global wireless communities. *Proceedings of the 1st ACM workshop on User-provided networking: challenges and opportunities*, pages 1–6, 2009.

[5] E.J Friedman and D.C Parkes. Pricing wifi at starbucks: issues in online mechanism design. *Proceedings of the 4th ACM conference on Electronic commerce*, pages 240–241, 2003.

[6] A Goldberg and J Hartline. Competitive auctions for multiple digital goods. *Algorithms—ESA 2001*, pages 416–427, 2001.

[7] Amin Mazloumian, Mohammad Manshaei, Mark Félegyházi, and Jean-Pierre Hubaux. Optimal pricing strategy for wireless social community networks. *NetEcon '08: Proceedings of the 3rd international workshop on Economics of networked systems*, Aug 2008.

[8] R.B Myerson. Optimal auction design. *Mathematics of operations research*, 6(1):58, 1981.

[9] S Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2009.

[10] J Shneidman and DC Parkes. Specification faithfulness in networks with rational nodes. *Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 88–97, 2004.

[11] J Shneidman and D.C Parkes. Specification faithfulness in networks with rational nodes. *Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 88–97, 2004.

[12] J Shneidman, D.C Parkes, and L Massoulié. Faithfulness in internet algorithms. *Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 220–227, 2004.

[13] Yoav Shoham and Kevin Leyton-Brown. Multiagent systems: Algorithmic, game-theoretic, and logical foundations. pages 1–532, Mar 2010.

[14] HR Varian and JK Mackie-Mason. Pricing the internet. *Public Access to the Internet*, 1994.

[15] HR Varian and JK Mackie-Mason. Pricing the internet. *Public Access to the Internet*, 1994.

[16] F Wu, T Chen, S Zhong, LE Li, and YR Yang. Incentive-compatible opportunistic routing for wireless networks. *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 303–314, 2008.

[17] A Zemlianov and G De Veciana. Cooperation and decision-making in a wireless multi-provider setting. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 1:386–397, 2005.

[18] S Zhong, J Chen, and YR Yang. Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, 3:1987–1997, 2003.