Abstract

Wireless spectrum is a valuable and scarce resource that currently suffers from under-use because of the dominant paradigm of exclusive-use licensing. We propose the `SATYA` *auction (Sanskrit for truth), which allows short-term leases to be auctioned and supports diverse bidder types, including those willing to share access and those who require exclusive-use access. Thus, unlike unlicensed spectrum such as Wi-Fi, which can be shared by any device, and exclusive-use licensed spectrum, where sharing is precluded,* `SATYA` *improves efficiency through supporting sharing alongside quality-of-service protections. The auction is designed to be scalable, and also strategyproof, so that simple bidding protocols are optimal. The primary challenge is to handle the externalities created by allocating shared-use alongside exclusive-use bidders. Using realistic Longley-Rice based propagation modeling and data from the FCC's CDBS database, we conduct extensive simulations that demonstrate* `SATYA`*'s ability to handle heterogeneous bidders involving different transmit powers and spectrum needs.*

0.1 Introduction

Currently, spectrum is licensed by governments in units covering large areas at high prices and for long periods of time, which creates a large barrier to entry for new applications. The main alternative, unlicensed bands such as Wi-Fi, has offered tremendous benefit, but is subject to a "tragedy of the commons" where these bands become congested and performance suffers [6].

Many researchers and firms (*e.g.,* [4, 35, 18]) have proposed creating a *secondary market* for dynamic spectrum access to provide a new way to access spectrum. The idea is that primary spectrum owners would be able to sell short-term leases. The same technology could also be used by the government to provide a new approach to the licensing of government owned spectrum, by selling short-term licenses in a primary market. This is made possible by recent advances in building spectrum registries [29, 18].

Rather than just sell exclusive-use access to spectrum, we advocate the adoption of auction technology to allocate spectrum to both exclusive-use and shared-use. An exclusive-use license guarantees a winner no interference but can be inefficient. For example, devices such as wireless microphones are only used occasionally, and other devices can use the same spectrum when the wireless microphone is not in use. This heterogeneity of devices and demand patterns presents an opportunity for sharing. In addition, many devices are capable of using a *medium access controller* (MAC) to share spectrum when there is contention.

Auctions generate revenue and also enable efficient spectrum use through the dynamic reprovisioning of spectrum. Auctions improve efficiency relative to fixed price schemes by allowing prices to adapt dynamically in response to varying demand. In addition, auctions provide incentives for different users to describe

through bids their access requirements, for example specifying exclusive-use or allowing sharing.

We describe the SATYA auction, which is designed to allocate short-term spectrum access across a wide range of scenarios, embracing different access technologies and different types of users, including individuals and service providers. This makes SATYA well suited to handle the mixture of users found today in settings such as Wi-Fi while still providing a better service than unlicensed spectrum. In determining the allocation of spectrum, SATYA considers the effect of interference on the value of the allocation to different bidders. Because of the possibility of sharing spectrum, bidders in SATYA care about how spectrum is allocated to other bidders, along with the dynamic access patterns of those bidders. Most existing auction designs for wireless spectrum either fail to allow sharing, or fail to scale to realistic problem sizes.

In order to make the algorithms for winner determination and determining payments scalable we impose structure on the bidding language with which bidders describe the effect of allocation to others on their value. The language allows bidders to express their value for different allocations, given probabilistic activation patterns, an interference model, and under different requirements expressed by bidders for shared vs exclusive-access spectrum. In determining the the value of an allocation, the auction must determine the fraction of each bidder's demand that is satisfied in expectation, considering sharing and interference patterns. For this purpose, we adopt a model for resolving contention by devices and assume knowledge of which devices will interfere with each other given allocation (based on device location), and represented through a conflict graph.

The SATYA auction is *strategyproof*, which is a property that makes simple bidding protocols optimal for users or the devices representing users. In particular, the utility-maximizing (the utility to a user is modeled as the difference between the user's value and the price) strategy is to bid truthfully, regardless of the bids, and regardless of the kinds of activation patterns and sharing or exclusive-use preferences of other users.

Strategyproofness is a property that is desirable for large-scale, distributed systems involving self-interested parties because it promotes stability— the optimal bid is invariant to changes in bids from other users. In comparison, bidders would need to keep changing their bids in a non-strategyproof auction in order to maximize utility. This continual churn in bids imposes an overhead on system infrastructure as well as participants.

Even without sharing, the problem of finding a value-maximizing (and thus efficient) allocation of spectrum is NP-hard [19]. In obtaining scalability, we adopt a greedy algorithm for determining the winners and the spectrum allocation. A crucial difficulty that arises because of externalities is that a straightforward greedy approach to allocation fails to be monotonic. What this means is that a user can submit a larger bid but receive less spectrum (in expectation, given the interference, sharing and activation patterns). Monotonicity is sufficient, and

essentially necessary, for strategyproof auctions [30]. This leads to a significant design challenge.

In recovering monotonicity, SATYA modifies the greedy algorithm through a novel combination of bucketing bids into value intervals in which they are treated equally (this idea was employed in Ghosh and Mahdian [12]), along with a "computational ironing" procedure that is used to validate the monotonicity of an allocation and perturb the outcome as necessary to ensure monotonicity (this idea was introduced by Parkes and Duong [31]). These techniques prevent cases in which an increase in bid can change the decision of the greedy algorithm to something that looks just as good given decisions made so far, but turns out to be worse for the bidder because of interference with other bids that are subsequently allocated.

In evaluating SATYA, we use real-world data sources to determine participants in the auction, along with the Longley-Rice propagation model [3] and high-resolution terrain information, to generate conflict graphs. We compare the performance of SATYA against other auction algorithms and baselines. Our results show that, when spectrum is scarce, allowing sharing through the SATYA auction increases efficiency by 40% over previous approaches while generating revenue for spectrum owners. The baseline also serves to provide an upper bound on the potential cost of requiring strategyproofness relative to a protocol that is designed to be efficient, but where participants in any case choose to behave in a way that is approximately truthful.

0.1.1 Related Work

Most proposed auction designs for the sale of short-term spectrum licenses preclude sharing amongst auction participants [5, 10, 33, 35, 36, 23]. From amongst these, VERITAS [35] was the first strategyproof design. However, VERITAS does not support sharing. We compare SATYA to VERITAS in the empirical analysis.

Ileri et al. [17, 16] consider models where users have exclusive access but only for short time periods, which effectively permits some amount of sharing. Kasbekar and Sarkar [24] propose a strategyproof auction and allow for sharing amongst winners. But the winner determination algorithm in their proposed auction is not scaleable because bids are explicitly represented on different joint spectrum allocations, which requires considering an exponential (in the number of participants) number of allocations. In contrast, our use of a structured bidding language allows us to achieve good efficiency while considering a polynomial number of allocations. Huang et al. [15] propose an auction design where bidders bid to share a single channel, but their design is not strategyproof and they do not address the issue of how to assign channels when multiple are available.

Gandhi *et al.* [11] propose an auction that allows sharing amongst winners, but differs from our approach in that it does not provide strategyproofness (and thus lacks an equilibrium analysis and is otherwise hard to evaluate), and precludes sharing between users who want exclusive-use when active but are only

intermittently active (e.g., wireless microphone devices) and other users. Externalities have also been considered in auction theory [21, 22], but without the combinatorial aspect of our allocation problem and the difficulties this implies for achieving strategyproofness. A number of papers have considered externalities in online advertising [8, 12, 13, 25, 32]. However, this work (and similarly that of Krysta *et al.* [26] on the problem of externalities in general combinatorial auctions) is not directly relevant, as the externalities in spectrum auctions have a special structure, of which `SATYA` takes advantage in order to achieve compact bid representations and scalable winner determination.

0.2 Challenges in Auction Design

In this section we describe in some more detail the challenges that arise when designing a spectrum auction that permits sharing. First, we discuss the general form of an auction and define strategyproofness. Second, we present a cenreal result due to Myerson [30] that provides a general framework for designing strategyproof auctions through the use of a monotone allocation rule. Finally, we introduce *reserve prices*, which are a standard approach to increasing the revenue from an auction.

In the simplest type of auction, a single item is for sale. Each bidder i has private information about his value $V_i > 0$. Let $B_i \geq 0$ denote the bid from bidder i. Each bidder receives an allocation $A_i \in \{0, 1\}$, where $A_i = 1$ if the bidder gets the item and 0 otherwise. Feasibility insists that $\sum_i A_i \leq 1$. Writing $B = (B_1, \ldots, B_n)$ for bids from n bidders, then we can write the allocation selected as a function $A(B) = (A_1(B), \ldots, A_n(B))$. Finally, each bidder makes some payment $P_i \geq 0$, that depends on the bids, so we write $P_i(B)$.

In a standard model, a bidder's utility, which captures his preference for the outcome of an auction, is

$$U_i(B) = V_i A_i(B) - P_i(B), \tag{0.1}$$

and represents the true value for the allocation minus the payment. There are many ways such an auction can be run.

One approach, known as a *first-price auction*, is that each bidder names a price and the bidder who bids the most wins the item and pays their bid with $P_i(B) = B_i$ for the winner. With perfect knowledge, a bidder should bid slightly more than the highest bid of other bidders (to a maximum of V_i), in order to pay as little as possible. Thus bidders try to anticipate how much others will bid, and bid accordingly. This gives a first-price auction high strategic complexity.

Another approach, due to Vickrey [34], is a *second-price auction*, where each bidder names a price and the bidder who bids the most wins the item. However, instead of paying the bid price, the payment is equal to the bid of the second highest bidder. In such an auction, a bidder has a simple strategy that is

(weakly) optimal no matter what: bid true value $B_i = V_i$. The Vickrey auction is strategyproof.

DEFINITION 1. *An auction defined with allocation rule A and payment rule P is strategyproof if*

$$U_i(V_i, B_{-i}) \geq U_i(B'_i, B_{-i}) \tag{0.2}$$

for all bid profiles B, all agents i, and all alternate bids B'_i, where $B_{-i} = (B_1, \ldots, B_{i-1}, B_{i+1}, \ldots, B_n)$ is the bid profile without agent i.

As explained in the introduction, a strategyproof auction is desirable because of the effect it has in simplifying bidding strategies and because of the overhead it removes from the infrastructure by precluding the need for bids to be continually updated as bids from others change.

But how to design such an auction in our setting? One thing to recognize is that the allocation will be much more complicated: analogous to an item is a channel $\times$ location (where the location depends on the location of the bidder's device and the channel is a range of frequencies) In addition to there being multiple items to allocate, there will be interference such that the value of an item depends on the other bidders allocated similar items. In particular, bidders that are geographically close to each other and are allocated the same channel will interfere with each other.

Part of the challenge is to describe a concise language to represent a bidder's value for different possible allocations. Another part of the challenge is to ensure that the allocation can be computed in polynomial time. The NP-hardness of the winner determination problem precludes a general auction design due to Vickrey, Clarke, and Groves [7, 14, 34], that would be strategyproof and efficient in our domain.

In achieving strategyproofness, an important property is that an allocation algorithm be *monotone*, which requires that $A_i(B_i, B_{-i})$, where $B_{-i} = (B_1, \ldots, B_{i-1}, B_{i+1}, \ldots, B_n)$, is weakly increasing in the bid of bidder i, fixing the bids of others, so that $A_i(B_i, B_{-i}) \geq A_i(B'_i, B_{-i})$ for $B_i \geq B'_i$.

THEOREM 1 (Myerson [30]). An auction is strategyproof if and only if for all bidders i, and fixed bids of other bidders B_{-i},

1 $A_i(B)$ is a **monotone** function of B_i (increasing B_i does not decrease $A_i(B)$), and
2 $P_i(B) = B_i A_i(B) - \int_{z=0}^{B_i} A_i(z, B_{-i}) \mathrm{d}z$.

Even beyond strategyproofness, monotonicity is still a worthwhile goal because it guarantees that participants attempting to optimize their bid will only increase the amount they receive and the amount they pay when they increase their bid.

In the case of an auction for a single good, the nature of monotonicity is simple: a bidder must continue to win the good when bidding a higher price. However, in our setting, winning a channel alone is not sufficient to make a bidder happy. In particular, if the channel is heavily used by others in a bidder's neighborhood

it may have little value. Thus, a bidder cares not only about whether or not he is allocated a channel, but also who else is allocated the same channel. The effects of the allocations of other bidders on the value of winning a good are known as *externalities*. This complicates the auction design because the allocation rule must be monotone not only in whether a bidder gets a channel, but also the amount of sharing that occurs on that channel. But once an allocation rule has been developed that is monotone in this sense, the auction can be made strategyproof through standard methods.

While determining the prices bidders pay requires computing an integral, in many cases this integral has a simple form. For example, in the (deterministic) single good case the allocation to a bidder, A_i, only takes on two values: 0 when the bidder does not get the good and 1 when he does. Since the allocation must be monotone, it is entirely determined by the *critical value* where it changes from 0 to 1. Thus, computing the integral reduces to the problem of determining the minimum bid that the bidder could have made and still been allocated.

In addition to strategyproofness, the proposed auction designs for the allocation of short-term licenses and spectrum sharing can be evaluated in terms of the twin goals of:

- *Allocative efficiency*: rather than maximize throughput or spectral efficiency, allocate resources to maximize the total utility from the allocation. Thus, in addition to traditional metrics we also report the total value from the allocations determined at the outcome of `SATYA`.
- *Revenue*: good revenue properties are important in order to provide an incentive for spectrum owners to participate in the market.

Efficiency is often held to be of primary importance when designing a marketplace because it provides a competitive advantage over other markets, and encourages participation by buyers. Maximizing revenue can be at odds with efficiency because it can be useful to create scarcity in order to boost revenue. One way to do this is to adopt a *reserve price*. We will examine the tradeoff between efficiency and revenue that can be achieved by adjusting the reserve price in `SATYA`.

0.3 The Model of Shared Spectrum and Externalities

0.3.1 User Model

In order to find opportunities to share among heterogeneous users (e.g., a user with a wireless device, or a TV station), we need a language to describe the requirements of each possible *type* of user.

Our model uses discrete intervals of time (called *epochs*), with auctions clearing periodically and granting the right to users to contend for access to particular

channels over multiple epochs. Thus our approach models participants who regularly want spectrum in a particular location over a period of time. Participants who wish to enter or leave need to wait until the next time the auction is run. The ultimate allocation of spectrum arises through random activation patterns of users and interference effects, and depends on specifics of the medium-access control (MAC) contention protocol. The effect of this MAC protocol is modeled within `SATYA` in determining the allocation.

The interference between users and their associated devices is modeled through a *conflict graph*, $G = (V, E)$, such that each user i is associated with a vertex ($i \in V$) and an edge, $e = (i, j) \in E$ exists whenever users i and j would interfere with each other if they are both active in the same epoch and on the same channel. Note that, for service providers such as TV stations, defining the conflict graph may be complex as it requires making decisions about the acceptability of interference over some portion of the served area.

We allow for both exclusive-use and "willing to share" users, where the former must receive access to a channel without contention from interfering devices whenever they are active, while the latter can still obtain value through contending for a fraction of the channel with other interfering devices.

We say that a channel is *free*, from the perspective of user i in a particular epoch, if no exclusive-use user j, who interferes with i and is assigned the right to the same channel as i, is active in the epoch.

Formally, we denote the set of user types T. Each type $t_i \in T$ is a tuple $t_i = (x_i, a_i, d_i, p_i, C_i, v_i)$, where:

- $x_i \in \{0, 1\}$ denotes whether the user requires *exclusive-use* of a channel in order to make use of it ($x_i = 1$) or willing to share with another user while both are active on the channel ($x_i = 0$).
- $a_i \in (0, 1]$ denotes the *activation probability* of the user: the probability that the user will want to use the channel, and be active, in an epoch.
- $d_i \in (0, 1]$ is the *fractional demand* of the channel that a user who is willing to share access requires in order to achieve full value when active.
- $p_i \geq 0$ denotes the per-epoch penalty incurred by the user when active and the assigned channel is not free. Both exclusive-use and non exclusive-use users can have a penalty.
- $C_i \subseteq \mathcal{C} = \{1, 2, \ldots\}$, where $\mathcal{C}$ is the set of channels to allocate, each corresponding to a particular spectrum frequency, denotes the channels that user i is able to use (the user is indifferent across any such channel.)
- $v_i \geq 0$ denotes the per-epoch value received by the user in an epoch in which it is active, the channel is free, and in the case of non exclusive-use types, the user receives at least a share d_i of the available spectrum.

In this model, each user demands a single channel. We discuss an extension to multiple channels in Section 0.4.6.

Some of the parameters that describe a user's type are a direct implication

of the user's technology and application domain. For example, whether or not a user requires exclusive-use when active and is unwilling or unable to share falls into this category. Users that can use a MAC will tend to be able to share, other users will tend not to be able to. As we explain below, users operating low-power TV stations or with wireless microphone devices would likely be in this category. The set of channels C_i on which a user's device can legally broadcast will tend to be easy to define.

For parameters such as the activation probability (how often the user makes use of the channel), and the fractional demand (how much of the channel is used when active), we assume that these can be estimated by the device, and then monitored by the network environment upon the outcome of an auction with the user punished if this information is mischaracterized. For example, a user could be banned from participating in future auctions. But certainly, the fractional demand d_i and activation probability a_i may be difficult to estimate in some cases, and especially when first bidding, due to uncertainty arising from the effects of interference, anticipated traffic, and propagation.

Examples

- A user who wishes to run a low-power (local) TV station on a channel would be unable to share it with others when active ($x_i = 1$), would be constantly broadcasting ($a_i = 1$), and would have a very large penalty p_i since it is unacceptable for the broadcast to be interrupted by someone turning on another (exclusive-use) device.
- A user with a wireless microphone cannot share a channel when active ($x_i = 1$), but is used only occasionally ($a_i = 0.05$) and has a smaller value of p_i since it may be acceptable if the user is occasionally unable to be used when there is another exclusive user also trying to use the channel.[1]
- A bidder may want to run a wireless network. Such a user would have constant traffic ($a_i = 1$), consume a large portion of the channel ($d_i = 0.9$), and might have a large penalty similar to a TV station for being completely disconnected. However, such a user is willing to share the channel with other non-exclusive types ($x_i = 0$), and will pay proportionately less for a smaller fraction of the bandwidth.
- A bidder representing a delay tolerant network [20], who occasionally ($a_i = 0.2$) would like to send a small amount of information ($d_i = 0.4$) if the channel is available. Such bidders might have a low or even no penalty as their use is opportunistic.

The per-epoch penalty is the cost to a user that is incurred in an epoch when the user is active (wants to use the network) but the channel is encumbered by an exclusive-use device. This can represent the known cost of using an alternate network or a contractual rebate.

[1] Indeed, it might make sense from an efficiency perspective to have several such devices share a channel if they interfere with each other sufficiently rarely.

The per-epoch value of a user v_i represents the dollar value that a user assigns to being able to access the channel when active, that is, in an epoch when the user wants to use the network. For an exclusive-use user, it is the per-epoch value for gaining exclusive access during that epoch. For a user willing to share, it is the per-epoch value for gaining a fraction d_i of the channel (as long as the channel is unencumbered by an exclusive-use device), and the assumption is that the value falls off linearly for a share below d_i. We will design a strategy-proof auction in which it is optimal for users to report their true value of v_i when bidding in the auction.

0.3.2 Allocation Model

Let $A_i \in C_i \cup \{\perp\}$ denote the channel allocated to each user i, where $\perp$ indicates the user has not been assigned a channel. Let $A = (A_1, \ldots, A_n)$ denote the joint allocation to n users. To allocate a channel means that the user has the right to contend for the channel when active, along with other users that interfere with the user and are allocated the same channel.

Exclusive-use users take priority over non exclusive-use users, and only experience interference when multiple exclusive-use users are simultaneously active. Non exclusive-use users share the channel when active simultaneously, and when the channel is free of exclusive-use users.

Let $V_i(A, t)$ denote the *expected value* to user i for allocation A given type profile $t = (t_1, \ldots, t_n)$. The value also depends on the conflict graph G, since this affects the interference between users. But we omit this term for notational simplicity.

An efficient allocation of spectrum maximizes the expected total value across the user population, that is

$$A^* \in \arg\max_A \sum_i V_i(A, t) \tag{0.3}$$

All allocations are feasible in our setting, since the expected value captures the negative externality due to interference. For this, we define the expected value $V_i(A, t)$ as,

$$= \begin{cases} 0 \text{ if } A_i = \perp, \text{ otherwise} \\ v_i \cdot a_i \Pr_i(F|A, t) \frac{E_A[S_i|F,t]}{d_i} - p_i \cdot a_i(1 - \Pr_i(F|A, t)). \end{cases} \tag{0.4}$$

A user's value depends first on the expected fraction of the user's request that can be satisfied. The user can only use the channel when it is not in use by another exclusive-use user, so we let $\Pr_i(F|A, t) \in [0, 1]$ denote the probability that the channel is *free* (F), with no exclusive-use user interfering with the allocated channel. Given that the channel is free, the user may still have to share with other users. For this, $E_A[S_i|F, t] \in [0, 1]$ denotes the maximum of the expected fraction of a channel that is available to user i given an epoch in which the channel is unobstructed by an exclusive-use user, the user is active, and given

user i's demand. For an exclusive-use user, this amount is always $E_A[S_i|F,t] = 1$, because such a user receives complete access to the channel when active and the channel is otherwise free.

Thus, the first term in (0.4) takes the expected fraction of channel capacity (necessarily less than d_i) supplied in an epoch in which the user is active, and in which the channel is free from exclusive-use users, and multiplies this by the probability the channel is free and the user is active $a_i \Pr_i(F|A,t)$, and the user's value for receiving d_i fraction of the channel in an epoch. This assumes that a user's value is linear in the available bandwidth (up to max-demand d_i.) The second term in (0.4) calculates the expected per-epoch *penalty* due to the channel not being free when a user is active (the probability of which is $a_i \cdot (1 - \Pr_i(F|A,t))$).

To complete this, we need to also define the probability that the channel allocated to user i is free, given allocation A and type profile t. This is given by the expression,

$$\Pr_i(F|A,t) = \prod_{j \in N_i \ s.t. \ A_i = A_j \wedge x_j = 1} (1 - a_j), \tag{0.5}$$

where N_i is the set of neighbors of i in G. This is the joint probability that no exclusive-use neighbor in the conflict graph, allocated the same channel as i, is active in an epoch.

Finally, we require an expression for $E_A[S_i|F,t] \leq d_i$, the expected fraction of a channel available to a user in an epoch when it is active and the channel is free. For this, we first consider the effect of a fixed number of active (non exclusive-use) neighbors in such an epoch.

For this, we assume a *Carrier Sense Multiple Access* (CSMA) style MAC, in which bandwidth is shared as equally as possible among active (non-exclusive-use) users, subject to the constraint that no user i receives more than its demand d_i. Formally, if N_a is a set containing i and the active neighbors of i with whom i shares a channel in the allocation, and $N_f = \{j \in N_a \mid d_j < f\}$, then user i receives a share of the available bandwidth on the channel equal to,

$$share_i(N_a, t) = \min \left(d_i, \max_{f \in [0,1]} \frac{1 - \sum_{j \in N_f} d_j}{|N_a - N_f|} \right) \tag{0.6}$$

The user either gets the full demand d_i or, failing that, the fair share (which the max in the equation determines). If all users have the same demand d_i, this reduces to each either the full demand being satisfied if $d_i \leq 1/|N_a|$ or receiving a $1/|N_a|$ share of the channel capacity otherwise. If some users demand less than their fair share, the remainder is split evenly among the others.

In completing an expression for $E_A[S_i|F,t]$, we adopt $\nu_i(A,c)$ to denote the set of neighbors of i on conflict graph G that, in allocation A, are allocated channel c. In particular, $\nu_i(A)$ denotes the set of neighbors allocated the same channel

as i. The probability that a particular set, $N' \subseteq \nu_i(A)$ is active in any epoch is,

$$active_i(N', t) = \left(\prod_{j \in N'} a_j \right) \left(\prod_{\ell \in \nu_i(A) - N'} (1 - a_\ell) \right) \tag{0.7}$$

From this, a user's expected share of the channel, given that the user is active and the channel is free (where the expectation is computed with respect to random activation patterns of interfering neighbors) is given by,

$$E_A[S_i|F, t] = \begin{cases} 0 & \text{if } \Pr_i(F|A, t) = 0 \\ 1 & \text{if } x_i = 1 \\ \sum_{N' \subseteq \nu_i(A)} active_i(N', t) share_i(N', t) & \text{, o.w.} \end{cases} \tag{0.8}$$

The two special cases cover exclusive-use users (who always receive their full demand when active, conditioned on the channel being otherwise free), and users for whom the channel is never free (for whom we arbitrarily define it to be 0, because the value in this case turns out to be irrelevant).

In general, computing $E_A[S_i|F, t]$ requires time exponential in the number of neighbors $\nu_i(A)$ with which i shares a channel. In making this practical, sharing can be limited to $d_{\max} \ll n$ neighbors, and the calculation can be completed in time that scales as $O(2^{d_{\max}})$. Alternatively, it may turn out that $d_{\max}$ is already small due to the nature of the conflict graph. Indeed, in our experiments for practical models of signal propagation, and even with hundreds of users participating in the auction, we did not need to impose such a limitation.

0.4 Auction Algorithm

Turning to the design of `SATYA`, we assume that the only component of a user's type that can be misreported is v_i, which represents the per-epoch value of the user when active. Designing an auction that is strategy-proof in regard to per-epoch value v_i is the focus of this section.

As explained in Section 0.3.1, it seems reasonable to assume that many of the parameters of a user's type can be checked by the network at the outcome of an auction and enforced through punishment (e.g., kicking out of the auction environment in the future), or are fundamental to the operation of a user's device (e.g., whether or not it can share the channel or needs exclusive-use) and thus not useful to manipulate. misreporting them (for example by remaining active and sending junk data to increase a_i), implementing such manipulations would be costly to users and in many cases would simply result in higher payments. Thus, for simplicity we ignore this possibility.

There are a number of interpretations of the penalty incurred by a user when a channel is encumbered by an exclusive-use device under which it seems reasonable to treat it as known. For example, the penalty could represent the cost

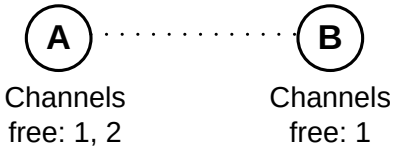


Figure 0.1 A potential violation of monotonicity. Nodes A and B are in contention range. At node A's location channels 1 and 2 are free; at B only channel 1 is free.

to gain access to a fallback network when this network is unavailable or a contractual payment the spectrum owner is obligated to make. The exact value may also not be important, as long as it is large. In practical terms, the value of the penalty controls the level of sharing that is permissible with exclusive users who are occasionally active, and getting this to the correct order of magnitude may be sufficient. In the extreme, `SATYA` works fine taking $p_i = 1$ for all i, which forbids such sharing.

Even if no users are permitted to share channels, finding an optimal allocation is NP-Hard [19]. Assigning bidders to channels such that no two neighbors have the same channel is a *graph coloring problem*. Therefore we adopt a greedy algorithm for allocation, modified to achieve monotonicity.

0.4.1 Externalities and Monotonicity

Let us first define *monotonicity* in our setting. For this, it is convenient to drop the complete type profile t from notation and write $\Pr_i(F|A)$ and $E_A[S_i|F]$ in place of $\Pr_i(F|A,t)$ and $E_A[S_i|F,t]$ respectively. In addition, let $b = (b_1, \ldots, b_n)$ denote the joint bid vector received from users, with $b_j \geq 0$ for all j.

Fixing the bids $b_{-i} = (b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n)$ of other users, an allocation algorithm $\mathcal{A}(b)$ (defining an allocation for every bid profile) is *monotone* if,

$$\begin{aligned}\Pr_i(F|\mathcal{A}(b_i', b_{-i}))E_{\mathcal{A}(b_i', b_{-i})}[S_i|F] \\ \geq \Pr_i(F|\mathcal{A}(b))E_{\mathcal{A}(b)}[S_i|F],\end{aligned} \tag{0.9}$$

for all bids $b_i' \geq b_i$. This insists that the *expected share* of a channel available to a user, conditioned on being active, weakly increases as the user's bid increases.

Figure 0.1 shows how monotonicity can fail for simple greedy algorithms. The greedy algorithm considers each user in (decreasing) order of bids and allocates the user to the best available channel in terms of maximizing value (or no channel if that is better). If there is a tie, the algorithm uses some tie-breaking rule, such as the lowest channel number. If user A has a lower bid than user B, the algorithm assigns user B to channel 1, then user A to channel 2, and both are fully satisfied. If user A raises its bid above that of user B, user A will be assigned to channel 1. Then, assuming sharing is better than leaving B unassigned, the algorithm assigns user B to channel 1, and user A receives less value due to interference.

0.4.2 High level approach

The monotonicity violation from Figure 0.1 would be prevented if the algorithm was not allowed to assign user B to channel 1 in the second case. We do this for many cases by assigning each user to a "bucket" based on his bid, such that the more a user bids the higher the bucket to which he is assigned. Users are not allowed to share with a user from a higher bucket. Thus, if user B is in a lower bucket than user A, user B will simply not be assigned a channel. If both users are in the same bucket, we will consider them in some order independent of their actual bids, and adopt in place of their bid value the minimal possible value associated with the bucket. The effect is that the allocation decision is invariant to a user's bid while the bid is in the same bucket. Since users are only allowed to share with other users within their buckets, the way buckets are chosen is an important parameter of our algorithm. Larger buckets create more possibilities for sharing. However, they also mean that the algorithm pays less attention to user's bids, so they may decrease the social welfare (the total value of the allocation) and revenue.

Bucketing prevents many violations of monotonicity, but it is not sufficient to prevent all of them. In particular, the example from Figure 0.1 can still occur if user A is in a lower bucket than user B and then raises his bid so they are in the same bucket (if he raises it to be in a *higher* bucket there is no problem). To deal with this case we adapt a technique known as "ironing"[31] to this domain. This is a post-processing step in which allocations that might violate monotonicity are undone. Given an input (a set of bids) to an allocation algorithm, the basic idea is to check the sensitivity of the allocation with respect to unilateral changes in the bid value by each bidder. In Parkes and Duong [31] this is applied to a problem of stochastic optimization, and a failure in regard to checking higher bids is addressed by unallocating the bidder at the current input. In the context of the `SATYA` auction, sensitivity is checked in regard to lower bids, and failure is addressed by unallocating other bidders that are sharing a channel with the user (improving the allocation for the bidder at the current input).

For each user allocated in the current bucket, we ask the counterfactual question "If this user were instead in the next lower bucket, is it possible he would be allocated?" If so, we guarantee that the user is satisfied in the current bucket by canceling (or "ironing") the allocations of other users with whom he shares. In Figure 0.1, if user A were in a lower bucket he would be allocated a channel. Therefore, in the ironing step, the algorithm would change user B's allocation and not allocate a channel in the current bucket. It will be important, though, that a channel allocation that is canceled in this way will be considered unavailable for future allocation. This prevents the need for nested arguments involving the effect of ironing on future allocations, future ironing of future allocations, and so on.

In this high level description, we have assumed that any two users who interfere with each other cannot share a channel without harming each other. In reality,

this is not the case; users capable of using a MAC and sending at sufficiently low rates will have a negligible effect on each other. Many of the more intricate details of our algorithm come from adapting the general approach to take advantage of this fact and allow more efficient use of wireless spectrum.

0.4.3 The SATYA Algorithm

`SATYA` begins by assigning each user i to a bucket based on the user's bid value b_i. There are many ways this can be done as long as it is monotone in the user's bid. For example, user i with an activity-normalized bid $a_i b_i$ could be assigned to value bucket with bounds $[2^\ell, 2^{\ell+1})$. To be general, we assume that bucketing of values is done according to some function $\beta(k)$, such that bucket k contains all users with (normalized) bids $a_i b_i$ in the range $[\beta(k), \beta(k+1))$.

Once users are assigned to buckets they are assigned channels greedily, in descending order of buckets. The order of assignment across users within the same bucket is determined randomly. Let K_i denote the bucket associated with user i. A channel c is considered to be available to allocate user i at some step in the algorithm, and given the intermediate allocation A, if,

- the channel c is in C_i;
- assigning i would not cause an externality to a neighbor from a higher bucket: for all $j \in N_i$, with $K_j < K_i$,

$$\sum_{\ell \in \{\nu_j(A,c) \cup \{i\}\}} d_\ell \leq 1 \tag{0.10}$$

- and, the combined demands of i and the neighbors of i from higher buckets assigned to c are less than 1:

$$d_i \quad + \sum_{j \in \nu_i(A,c), K_j > K_i} d_j \leq 1 \tag{0.11}$$

We refer to the second condition as requiring that the demands of each neighbor of user i from a higher bucket be satisfied. The third condition requires that the demand of user i is satisfied. This does not preclude allocations where some user has $E[S_i|F,t] < d_i$. It simply requires that, in such cases, the user is sharing with others in the user's own bucket.

Suppose i is the next user to be considered for allocation. `SATYA` will identify the channel for which assigning i to the channel has the maximum marginal effect on the total value of all currently allocated users along with user i itself. To do so, for every channel c that is available to the user, and including $\perp$ (and thus not allocating any spectrum to the user), `SATYA` estimates the expected value to some user j after assigning i to c as

$$e_j(A,b) = \beta(K_j)\mathrm{Pr}_j(F|A,b)\frac{E_A[S_j|F,b]}{d_j} - a_j \cdot p_j(1 - \mathrm{Pr}_j(F|A,b)) \tag{0.12}$$

This estimate differs from the user's actual bid by assuming that each user

in a given bucket shares the same value. This is important for achieving monotonicity, because we need to ensure the decision for a user depends on the bucket associated with a user's bid value and not in more detail on a user's value.

Given this, user i is assigned to the channel that maximizes the sum of the expected bid values of each user already allocated and including its own value, and without leaving any user with a negative expected value. The optimal greedy decision might allocate $\perp$ to user i, and thus no spectrum. In the event of a tie, the user is assigned to the lowest numbered among the tied channels (including preferring $\perp$, all else equal).

After all users in a bucket are assigned channels, there is an *ironing* step in which monotonicity of the allocation is verified, and the allocation perturbed if this fails. Recall that monotonicity violations occur when the greedy allocation makes a "bad" decision for the user and would make a better one had the user been considered later. Bucketing prevents users from being able to move themselves later while staying in the same bucket, but they could still lower their bid enough to drop into the next bucket. To rule out this possibility, the ironing procedure re-runs the allocation procedure for each user with the user placed instead in the next lower bucket. If this counterfactual shows that the final allocation would be *better* for the user, then there is a potential monotonicity violation, and the provisional allocation is modified by changing the assignments of the neighbors with whom the user shared a channel to $\perp$. Checking only the next bucket is sufficient because if the user can be assigned in any lower bucket he can be assigned in the next bucket.

The complete algorithm is specified in pseudocode as Algorithm 1. In the specification, we use distinct names to be able to refer to allocations created along the way. The variable $A(k, i, j)$ denotes the state of the allocation in bucket k after considering the jth user in the order given by random permutation π on users. Some of these allocations will be used for the counterfactual questions asked by ironing, so i is the user currently being omitted ($i = 0$ if there is no such user).

LEMMA 2. Algorithm 1 is monotone.

The proof of Lemma 2 is presented in the appendix.

0.4.4 Pricing Algorithm

Given a monotone allocation algorithm, then the payment to collect from each user is defined as is standard from Myerson [30]. As we saw with the single good setting in Section 0.2, we can exploit the structure of our allocation rule to compute the required integral. Because of the way ironing works, there is exactly one bucket in which a user can receive an allocation in which the user shares a channel with other users. In any lower bucket, the user does not get allocated a channel; in any higher bucket the user is guaranteed by ironing to have the user's demand fully satisfied in the allocation. Thus there are only three possible

Algorithm 1 `SATYA` Allocation Algorithm

$\pi \leftarrow$ a random permutation of $1 \dots n$
$M \leftarrow \max_i K_i$
$m \leftarrow \min_i K_i$
$Allocation_i \leftarrow \perp \forall i$
$A_i(M+1, 0, n) \leftarrow \perp \forall i$
// Do Provisional Allocation
for $k = M$ to m by -1 **do**
  $A(k, 0, 0) \leftarrow A(k{+}1, 0, n)$ // Initialize allocation to result of previous bucket
  for $j = 1$ to n **do**
    $A(k, 0, j) \leftarrow A(k, 0, j-1)$
    if $K_{\pi(j)} = k$ **then**
      $c \leftarrow \text{AssignChannel}(A(k, 0, j), \pi(j))$ // $\pi(j)$ in bucket so assign him
      $A_{\pi(j)}(k, 0, j) \leftarrow c$
      $Allocation_{\pi(j)} \leftarrow c$
// Counterfactuals to use for ironing
for $i = 1$ to n **do**
  $A(K_i, i, 0) \leftarrow A(K_i{+}1, 0, n)$ // Prepare to reallocate i's bucket without him
  for $j = 1$ to n **do**
    $A(K_i, i, j) \leftarrow A(K_i, i, j-1)$
    if $K_{\pi(j)} = K_i \wedge \pi(j) \neq i$ **then**
      $A_{\pi(j)}(K_i, i, j) \leftarrow \text{AssignChannel}(A(K_i, i, j), \pi(j))$
// Do ironing
for $i = 1$ to n **do**
  $free \leftarrow \exists$ avail. c for $\pi(i)$ given $A(K_{\pi(i)}, \pi(i), n)$ // Does i trigger ironing?
  if $Allocation_{\pi(i)} \neq \perp \wedge free$ **then**
    $nbrs \leftarrow \nu_{\pi(i)}(Allocation)$ // Cancel neighbors until i is happy
    while $d_{\pi(i)} + \sum_{j \in nbrs} d_j > 1$ **do**
      $j \leftarrow$ last $j \in nbrs$ according to π
      $Allocation_j \leftarrow \perp$
      $nbrs \leftarrow nbrs - \{j\}$
return *Allocation*

AssignChannel(A, i):
$channels \leftarrow \{c$ available for i given $A\}$
for all $c \in channels \cup \{\perp\}$ **do**
  $A_i \leftarrow c$
  $value_c = \sum_{j=1}^{n} e_j(A, b)$ // Calculate estimated social welfare
  if $\exists j$ s.t. $e_j(A, b) < 0$ **then**
    $value_c = 0$ // Do not give anyone negative utility
return $\arg\max_c value_c$ (break ties in favor of $\perp$, then lowest channel number)

allocations the user might obtain as the bid value of the user changes and the relevant critical values are determined by finding the lowest bucket in which the user would be allocated and computing how much he values that allocation. Algorithm 2 shows how the associated bucket can be determined, and what price should be charged in each case.

Algorithm 2 Pricing Algorithm

$M \leftarrow \max_i K_i$
$m \leftarrow \min_i K_i$
for $i = 1$ to n **do**
  if $Allocation_i = \perp$ **then**
    $P_i = 0$
  else
    run Algorithm 1 without user i to get $A'(k, 0, n) \forall k$
    $k = M$
    while $k > m - 1 \wedge \exists c \in C_i$
     s.t. c is available in $A'(k, 0, n)$ **do**
      $k = k - 1$
    // k is now the unique bucket in which i might share
    run Algorithm 1 with i in bucket k to get $Allocation'$.
    $f \leftarrow \Pr_i(F|Allocation')$
    $s \leftarrow E_{Allocation'}[S_i|F]$
    if $K_i > k$ **then**
      $P_i \leftarrow \beta(k+1) - (\beta(k+1) - \beta(k))fs$
    else
      $P_i \leftarrow \beta(k)fs - p_i(1-f)$

THEOREM 3. `SATYA` is strategyproof with respect to bid value.

This result follows because `SATYA` allocates channels using Algorithm 1 and charges payments according to Algorithm 2, which are the correct "Myerson" payments. While the result follows from Myerson, a slight modification is needed because in our model a user's utility depends on the penalty p_i in a way that makes them not quite fit the definition of a single-parameter domain. For this reason, we provide a direct proof of strategyproofness in the appendix.

0.4.5 Running time

Recall that n is the number of users, and let $\chi = |\mathcal{C}|$ denote the number of channels. (in general $\chi < n$ or the algorithm can assign each user his own channel). The running time of `SATYA` is determined largely by the implementation of the AssignChannel procedure. As discussed in Section 0.3.2, this require computation that scales exponentially in the number of neighbors with which i shares each channel considered. Thus, by in domains where this is limited to at most

r neighbors then the call to AssignChannel requires time $O(\chi n 2^r)$. Indeed, we did not need to impose any limit on the number of neighbors in generating our simulation results, because users' utilities were such that it did not make sense for users to share with a large number of other users.

THEOREM 4. SATYA's running time is at most $O(\chi n^4 2^r)$, where n is the number of users, χ is the number of channels, and r is the maximum number of sharing neighbors considered.

As the proof shows, the running time is dominated by the time needed for $O(n^3)$ calls to AssignChannel.

Proof SATYA needs to calculate $A(k, 0, n)$ for each non-empty bucket k and $A(K_i, i, n)$ for each user i. There are at most n non-empty buckets and n users for a total of $2n$ allocations to be computed. Each allocation requires assigning a channel to each user at most once, so there are $O(n^2)$ calls to AssignChannel. Ironing takes time $O(\chi n)$ per user for a total of $O(\chi n^2)$, so the running time of the allocation is dominated by the calls to AssignChannel (which needs at least time χ to consider each channel).

The pricing algorithm runs for each user and runs the allocation algorithm twice: once to determine in which bucket the user might share and once to determine what the user's share would be in that bucket. Thus SATYA requires $2n+1$ runs of the allocation algorithm for a total of $O(n^3)$ calls to AssignChannel. □

0.4.6 Extensions

An earlier auction proposal, VERITAS [35], suggests a number of ways to handle assignments of a user to multiple channels. In particular, users can either require a specific number of channels or be willing to accept a smaller number than they request. Users may also wish to insist that an allocation of multiple channels be contiguous. SATYA can be extended to allow all of these. We omit discussion of the algorithmic changes required, but we present simulation results in Section 0.5.4. Essentially, these changes require appropriately adapting the notion of when a group of channels is "available" to a user.

SATYA has a number of parameters. One obvious choice is the function β, which is used to assign users to buckets. Any function that is monotone in a user's bid can be used. This includes functions that take into account other facts about the user, for example the user's type or the number of neighbors the user has in the conflict graph. Another area of flexibility in defining SATYA is in the role of the permutation π. Rather than a random perturbation, any method that does not depend on user bids can be used. Some natural possibilities include ordering users by their degree in the conflict graph (so that users who interfere less are allocated first), ordering by a combination of activation probability and demand (so that users who use less spectrum are allocated first), or considering exclusive-

User Type	*Act. Prob.*	*Value*	*Penalty*	*Demand*
Exclusive-Continuous	1	[0, 1000]	10000	1
Exclusive-Periodic	[0.05, 0.15]	[0, 1000]	5000	1
Sharing-High	1	[0, 1000]	10000	[0.3, 1]
Sharing-Low	[0, 1]	[0, 1000]	5000	[0.3, 1]

Table 0.1 Mix of user types used in the evaluation

use users last since they impose much larger externalities on those with whom they share. We leave further exploration of this direction for future work.

0.4.7 SATYA's use of a MAC

As mentioned in Section 0.3.2, we use a simple model to calculate what happens when users share a channel. Our simple model can be replaced by a more sophisticated model from prior work that has explored the capacity of CSMA based wireless networks (*e.g.,* [27, 38, 37, 28]) as long as, in expectation, having more neighbors decreases a user's share of the channel. This model can also be extended in other interesting ways. For example, we could add for each user i a parameter ℓ_i, such that if he receives less than an ℓ_i fraction of the channel it is useless. This simply requires defining the share to be 0 if it would be less than ℓ_i. Alternatively, we could adopt TDMA rather than CSMA.

For implementation, SATYA does not require drastic changes to existing MACs. The primary requirement is for a user to stop transmitting when it is another user's turn (in the case of exclusive-use users). This is not unique to SATYA and is, for example, required of devices that use white spaces. However, a small change is required to a user's network stack to seek to transmit only when the user wins the auction (and therefore is allowed to contend for a channel). This can be implemented anywhere in the software stack.

0.5 Evaluation

In this section we compare the performance of SATYA to VERITAS. Since VERITAS does not permit sharing, we modify it slightly and implement VERITAS-S, which permits sharing as long as there are no externalities imposed (i.e. sharing is permitted only when the combined demands of users that wish to share do not exceed the capacity of the channel). We also implement GREEDY, a version of SATYA without bucketing and ironing that provides higher overall efficiency. *GREEDY is neither strategyproof nor monotone.* Thus, bids need not match their true values. However, to set as high a bar as possible, we assume they do so. Since it gets to act on the same information but has fewer constraints than SATYA, GREEDY serves as an upper bound for our experiments.

Parameters: As shown in Table 0.1, all our experiments use four classes of user types bidding for spectrum. Note that, in the table, we we have normalized

the values so the table reflects the range of $a_i v_i$ rather than the range of v_i. Each class represents different applications. For example, a TV station serving a local community is a user who wants exclusive access for a long period of time. A wireless microphone is an example of a user who wants exclusive access but for short periods of time. A low-cost rural ISP is an example of a *Sharing-High* user who expects to actively use the spectrum but can potentially tolerate sharing, and a regular home user is an example of a *Sharing-Low* user whose spectrum access pattern varies. Note, each class of users may have different transmit powers and coverage areas than the others. Since our goal is to evaluate the efficacy of `SATYA` in exploiting opportunities for sharing, we assign 5% of the total users as exclusive-continuous, 15% exclusive-shared, 30% Sharing-High, and the remaining 50% Sharing-Low.

Methodology: Each auction algorithm takes as input a conflict graph for the users. To generate this conflict graph in a realistic manner, we implement and use the popular Longley-Rice [2] propagation model in conjunction with high resolution terrain information from NASA [1]. This sophisticated model estimates signal propagation between any two points on the earth's surface factoring in terrain information, curvature of the earth, and climactic conditions. We use this model to predict the signal attenuation between users, and consequently the conflict graph.

We use the FCC's publicly available CDBS [9] database to model the transmit power, location, and coverage area of Exclusive-Continuous users. Note, that this information as well as the signal propagation predictions are sensitive to geographic areas.

We model the presence of all other types of users using population density information. Users are scattered across a 25 mile x 25 mile urban area in a random fashion by factoring in population density information. Since each class of user has a different coverage area, we determine that a pair of nodes conflicts if the propagation model predicts signal reception higher than a specified threshold.

We repeat each run of the experiment 10 times and present averaged numbers across runs. Unless otherwise specified, the number of channels is 5. In tuning `SATYA`, we experimented with a variety of methods for determining to which bucket to assign a user. On the basis of this analysis we adopt buckets of size 500 ($\beta(k) = 500k$).

In our experiments, we use the following metrics:

- *Allocated Users*: The total number of users allocated at least one channel by the auction algorithm.
- *Efficiency*: The sum of the valuations for the allocation by allocated users including the effect of any interference and preemption.
- *Satisfaction:* The sum of the fraction of a user's total demand that is satisfied over all users.
- *Spectrum Utilization*: The sum of satisfaction weighted by activation probability and demand. From a networking perspective, spectrum utilization is a

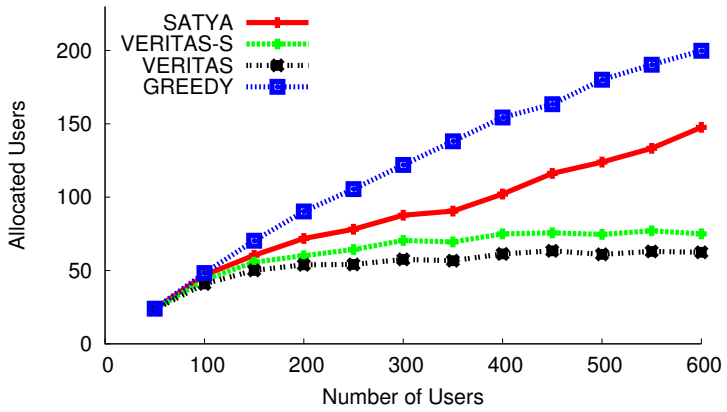


Figure 0.2 Number of users allocated spectrum, as a function of the number of users participating in the auction.

measure of how much the spectrum is being used (similar to the total network capacity).

- *Revenue*: The sum of payments received from users.

0.5.1 Varying the Number of Users

Figure 0.2 and Figure 0.3 show the performance of various algorithms as a function of the number of users participating in the auction. As we vary the number of users, we keep the mix of user types to be the same as Table 0.1.

As seen in Figure 0.2, as the number of users increases, `SATYA` produces up to 72% more allocated users when compared to VERITAS and VERITAS-S. This gain comes from being permitted to allocate users despite the presence externalities. With fewer users, all three algorithms demonstrate similar performance because almost all users can either be allocated a channel of their own or are impossible to satisfy.

Overall, VERITAS-S and VERITAS do not make the best use of users that can share. This is demonstrated in Figure 0.3, which is the distribution of different classes of users assigned channels by each algorithm. As the number of users increases, VERITAS-S and VERITAS significantly reduce the fraction of users capable of sharing who are assigned channels (relative to `SATYA`). However, all algorithms demonstrate a similar performance in the fraction of exclusive bidders who are assigned channels. Hence, `SATYA` is capable of taking advantage of sharing by allocating channels to more of such users. As expected GREEDY outperforms all strategyproof auctions and is able to assign more sharing users. Although we omit a detailed presentation, the difference in performance between `SATYA` and GREEDY is primarily due to bucketing. Ironing does occur but has only a minor effect.

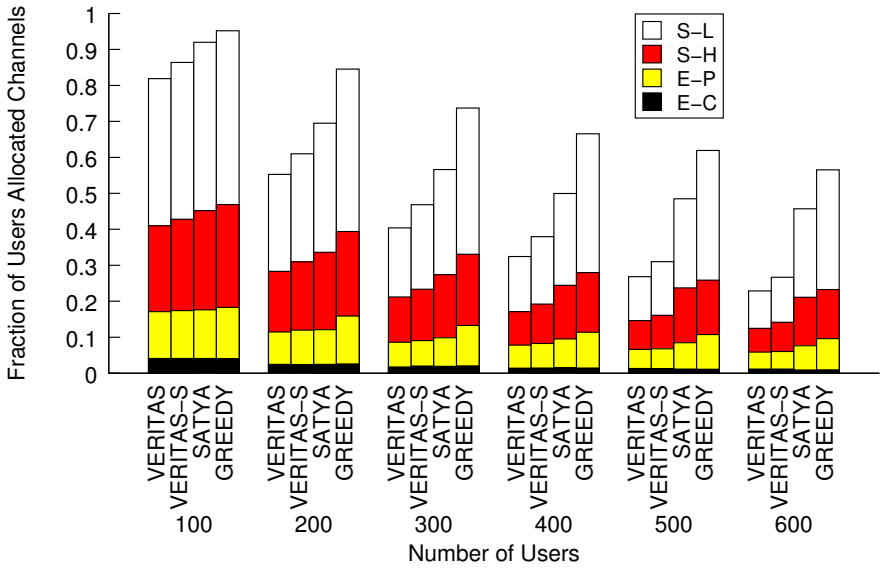


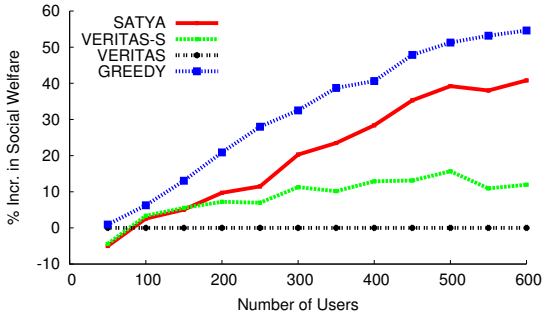
Figure 0.3 Distribution of user types across winning users, as the number of bidding users are varied. S-L are Sharing-Low users. S-H are Sharing-High users. E-P are Exclusive-Periodic users and E-C are Exclusive-Continuous users.

In addition to the number of users allocated spectrum, the results for other metrics are shown in Figure 0.4, which plots the results in terms of percentage improvement over the baseline of VERITAS. As seen in Figure 0.4(a), the relative efficiency attained by `SATYA` increases with an increase in the number of users. This is a direct consequence of assigning channels to more users capable of sharing the spectrum. This shows that, despite externalities from sharing, the additional users allocated consider it valuable. At 600 bidders, `SATYA` realizes a gain of 25% over VERITAS-S and 40% over VERITAS in terms of efficiency. Similarly, as seen in Figure 0.4(b), we find a 50% increase in the spectrum utilization of the network using `SATYA`. As efficiency, spectrum utilization, and satisfaction all take into account externalities, Figures 0.4(a), 0.4(b), and 0.4(c) show significant correlation. As with the users allocated metric, at fewer nodes the algorithms are indistinguishable as there are few opportunities to share.

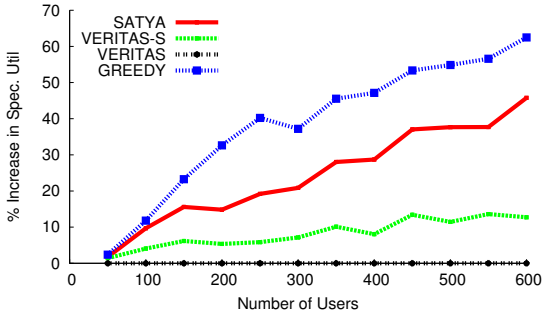
Hence, the main takeaway is that, `SATYA` increases the number of allocated users as well as efficiency.

0.5.2 Varying the Number of Channels

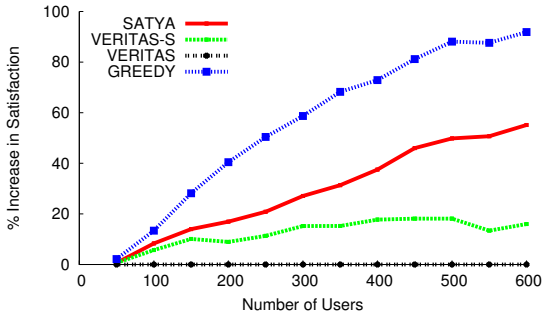
We also measure the effect of varying the number of channels auctioned on the overall outcome of the auction. The results shown in Figure 0.5 demonstrate the following trend: as the number of auctioned channels increases the gap in performance among the algorithms reduces. This is similar to having fewer bid-



(a) Efficiency



(b) Spectrum Utilization

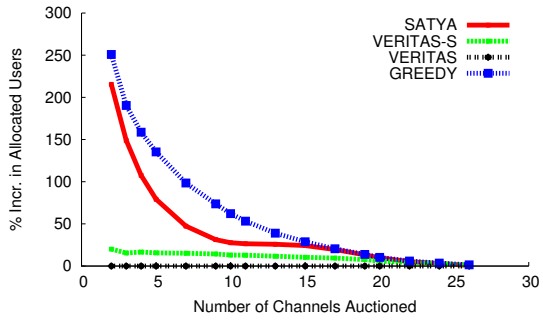


(c) Satisfaction

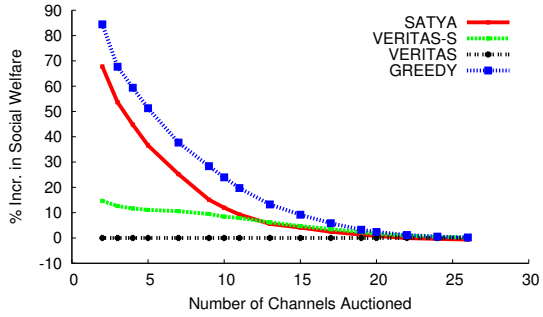
Figure 0.4 Effect of varying the number of users in the auction (compared to VERITAS-S, VERITAS, and GREEDY).

ders participate in the auction; with more channels, there is a reduced need for sharing and all algorithms perform similarly. As Figure 0.5(a) shows, `SATYA` is still able to assign more bidders than other algorithms until about 20 auctioned channels. Similarly, in Figure 0.5(b), we see that `SATYA` outperforms VERITAS by 20-60% in terms of efficiency up until about 10 channels.[2]

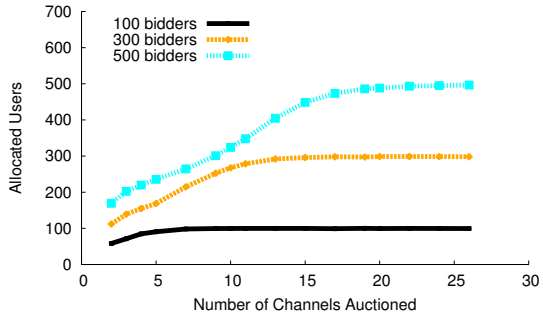
[2] We omit graphs for spectrum utilization and satisfaction for this and later experiments because they demonstrate a similar trend.



(a) Allocated Users



(b) Efficiency



(c) Allocated Users for `SATYA`

Figure 0.5 Effect of varying the number of channels auctioned (compared to VERITAS-S, VERITAS, and GREEDY).

We also vary the number of users and the number of channels simultaneously and the results for `SATYA` are shown in Figure 0.5(c). We see that as the number of users increases, `SATYA` takes advantage of the increased opportunity for sharing and allocate more users.

Hence, the main takeaway is `SATYA` *provides substantial benefits when the number of channels makes spectrum scarce.*

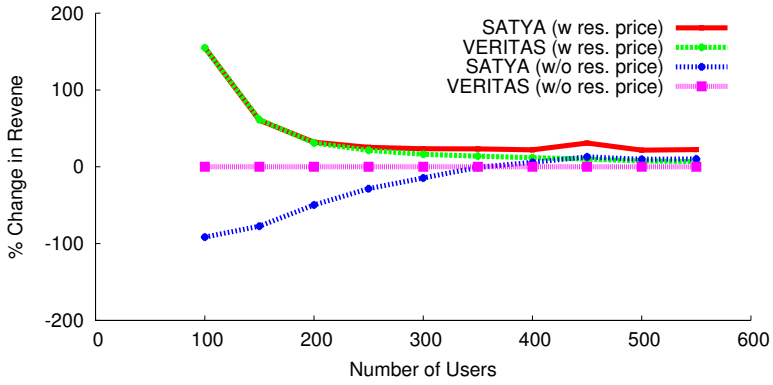


Figure 0.6 Impact of revenue, as a function of number of users.

0.5.3 Measuring Revenue

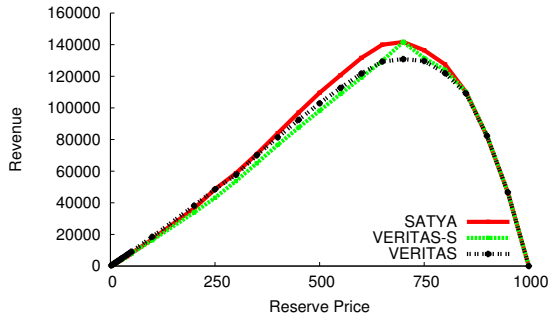
We consider efficiency the most important measure of performance: a market that finds success in the long run will allocate resources to those that find the most value. However, revenue is also important in providing sufficient incentive for spectrum owners to bring inventory to the auction.

First, we measure the total revenue obtained as a function of the number of users bidding for spectrum without reserve prices. We do not include GREEDY in this analysis because it is not strategyproof and it is not clear what users will bid and thus what the actual revenue would be. As seen in Figure 0.6, the revenue obtained by `SATYA` and is much lower than VERITAS for smaller numbers of users. We omit VERITAS-S from the figure for readability, but its performance also suffers. This is a consequence of sharing making it easier to accommodate users.

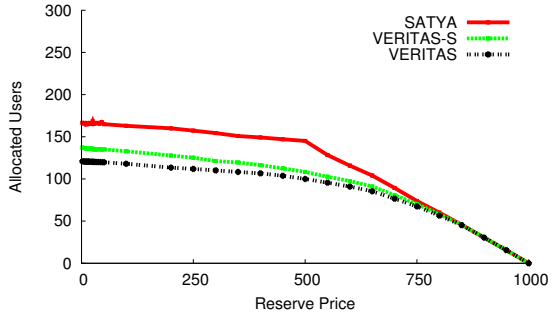
To improve revenue, we institute reserve prices.[3] While Myerson's approach in principle allows us to compute the optimal reserve price given knowledge of a distribution on values [30], we instead determine a suitable reserve price through simulation. The results from a simulation that varies the reserve prices is shown in Figure 0.7 for 300 bidding users. Figure 0.7(a) shows that with a reserve price of 0 (i.e. no reserve price) VERITAS performs better than `SATYA` and VERITAS-S in terms of revenue. As the reserve price begins to increase, the revenue derived from all three auctions increases. However, at around a price around 700 (depending on the algorithm), there is an inflection point in the revenue. As seen in Figure 0.7(b), this is because significantly fewer users are allocated by the auction and efficiency decreases (Figure 0.7(c)).

Based on these results, we adopt a reserve price of 400 and repeat the experiment to measure revenue by varying the number of bidders. We used a fixed reserve price for consistency; in practice it could depend on the number of users and

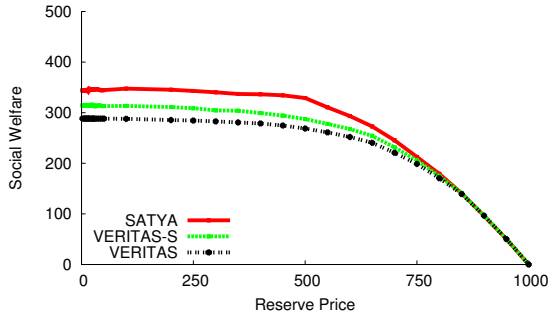
[3] VERITAS explored a similar opportunity to increase revenue by limiting the number of channels available.



(a) Revenue



(b) Allocated Users



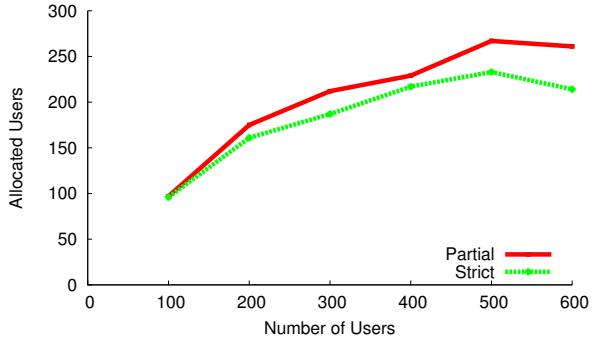
(c) Efficiency

Figure 0.7 Effect of reserve prices with 300 users on revenue, users allocated spectrum, and efficiency.

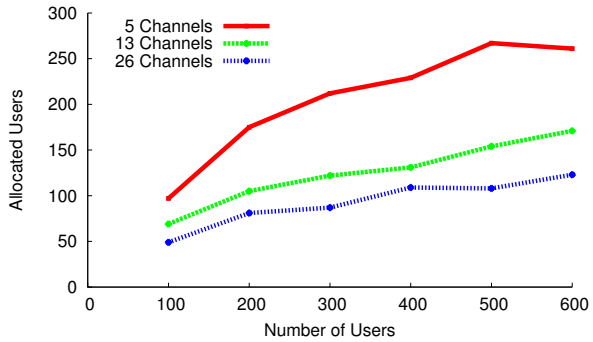
be individualized for each user. As Figure 0.6 shows, this increases revenue for the auctioneer significantly for all algorithms. The increase is most pronounced with 50 users (not shown because the improvement is so large) where revenue goes from essentially zero to approximately ten thousand. `SATYA`, which without a reserve price lost revenue by being too efficient in allocating users, benefits slightly more than VERITAS. With a large number of users, the reserve price is

essentially irrelevant because of the amount of competition; with 550 users the gain is below 12%.

0.5.4 SATYA's Performance with Multiple Channels



(a) Strict vs partial demands (5ch)



(b) Varying user demands

Figure 0.8 Experiments with multiple channels

SATYA is also capable of allowing users to bid for multiple channels in the auction. To illustrate this, we ran an experiment where we varied the number of channels that each user bids for as well as the number of users in the auction. Not all users bid for the same number of channels. The number of bid for is what a user with $d_i = 1$ would request; lower d_i results in a proportionally lower request. We used two modes of channel allocation schemes in SATYA, *strict*: when a user either gets the number of channels it requests for or nothing, and *partial:* a user can get fewer than requested channels. The total number of channels auctioned (not to be confused with the number of channels bid) was fixed to 26. The results are shown in Figure 0.8. As seen in Figure 0.8(a), partial allocations result in slightly more allocated users than strict, which is what we would expect since strict allocations are constraints that are harder to satisfy. Figure 0.8(b)

shows that increasing the the number of channels demanded by users reduces the number of winners as would be expected.

0.6 Conclusions

The SATYA auction is designed to allocate short-term spectrum licenses and allows for both bidders who are willing to share as well as bidders who require exclusive-use when active. SATYA does this while still allowing for quality-of-service guarantees. From a technical perspective, introducing sharing introduces allocative externalities and the auction algorithm is designed to handle these in a way that preserves strategy-proofness. Using realistic simulations, we showed that the ability of SATYA to share spectrum results in superior allocations by a variety of metrics. Our simulations also showed that the costs of achieving strategyproofness in this setting are minimal relative to the efficiency of a greedy but non strategyproof algorithm, and we believe that strategy-proofness is a price worth paying for the resulting improvements in simplicity and stability.

References

[1] "Shuttle Radar Topograph Mission (SRTM), http://www2.jpl.nasa.gov/srtm/."
[2] "The ITS Irregular Terrain Model Algorithm, NTIA, Department of Commerce."
[3] "OET BULLETIN No. 69, Longley-Rice Methodology for Evaluating TV Coverage and Interference," February 2004. [Online]. Available: http://www.ieee.or.com/Archive/uwb.pdf
[4] M. M. Buddhikot, P. Kolodzy, S. Miller, K. Ryan, and J. Evans, "DIMSUMNet: new directions in wireless networking using coordinated dynamic spectrum access," in *Conf. on a World of Wireless, Mobile and Multimedia Networks*, 2005.
[5] M. M. Buddhikot and K. Ryan, "Spectrum management in coordinated dynamic spectrum access networks," in *DySPAN*, 2005.
[6] M. M. Bykowsky, M. A. Olson, and W. W. Sharkey, "Modeling the efficiency of spectrum designated to licensed service and unlicensed operations," FCC OSP Working Paper Series, Tech. Rep., 2008.
[7] E. Clarke, "Multipart pricing of public goods," *Public Choice*, vol. 8, pp. 17–33, 1971.
[8] F. Constantin, M. Rao, C.-C. Huang, and D. C. Parkes, "On Expressing Value Externalities in Position Auctions," in *Proc. of the 6th Ad Auctions Workshop*, 2010.
[9] FCC Media Bureau, "TVQ TV Database, http://www.fcc.gov/mb/video/tvq.html."
[10] S. Gandhi, C. Buragohain, L. Cao, H. Zheng, and S. Suri, "A general framework for wireless spectrum auctions," in *DySPAN*, 2007.
[11] ——, "Towards real-time dynamic spectrum auctions," *Computer Networks*, vol. 52, no. 4, pp. 879–897, 2008.
[12] A. Ghosh and M. Mahdian, "Externalities in online advertising," in *17th International World Wide Web Conference (WWW)*, 2008.

[13] R. Gomes, N. Immorlica, and E. Markakis, "Externalities in keyword auctions: An empirical and theoretical assessment," in *WINE*, 2009, pp. 172–183.

[14] T. Groves, "Incentives in teams," *Econometrica*, vol. 41, no. 4, pp. 617–631, 1973.

[15] J. Huang, R. A. Berry, and M. L. Honig, "Auction mechanisms for distributed spectrum sharing," in *Proc. of 42nd Allerton Conf.*, 2004.

[16] O. Ileri, D. Samardzija, and N. B. Mandayam, "Dynamic Property Rights Spectrum Access: Flexible Ownership Based Spectrum Management," in *DySpan*, 2007.

[17] O. Ileri, D. Samardzija, T. Sizer, and N. B. Mandayam, "Demand Responsive Pricing and Competitive Spectrum Allocation via a Spectrum Server," in *DySpan*, 2005.

[18] S. B. Inc., "The secondary spectrum market: A licensing & leasing primer."

[19] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," *Wireless Networks*, vol. 11, no. 4, pp. 471–487, 2005.

[20] S. Jain, K. Fill, and R. Patra, "Routing in a Delay Tolerant Network," in *SIGCOMM*, 2004.

[21] P. Jehiel, B. Moldovanu, and E. Stacchetti, "How (not) to sell nuclear weapons," *The American Economic Review*, vol. 84, no. 6, pp. 814–829, Sep. 1996.

[22] ——, "Multidimensional mechanism design for auctions with externalities," *Journal of Economic Theory*, vol. 85, pp. 258–293, 1999.

[23] J. Jia, Q. Zhang, Q. Zhang, and M. Liu, "Revenue generation for truthful spectrum auction in dynamic spectrum access," in *Proc. of the 10th ACM Int. Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2009, pp. 3–12.

[24] G. S. Kasbekar and S. Sarkar, "Spectrum auction framework for access allocation in cognitive radio networks," in *Proceedings of the 10th ACM Interational Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2009, pp. 13–22.

[25] D. Kempe and M. Mahdian, "A cascade model for externalities in sponsored search," in *Internet and Network Economics, 4th International Workshop (WINE)*, 2008, pp. 585–596.

[26] P. Krysta, T. Michalak, T. Sandholm, and M. Wooldridge, "Combinatorial auctions with externalities," in *Ninth International Conference on Autonomous Agents And Multiagent Systems (AAMAS)*, 2010.

[27] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris, "Capacity of Ad Hoc Wireless Networks," in *MOBICOM*, 2001.

[28] V. Mhatre and K. Papagiannaki, "Optimal Design of High Density 802.11 WLANs," in *CoNEXT*, 2006.

[29] R. Murty, R. Chandra, T. Moscibroda, and P. Bahl, "SenseLess: A Database-Driven White Spaces Network," in *IEEE DySpan*, 2011.

[30] R. Myerson, "Optimal auction design," *Mathematics of Operations Research*, vol. 6, pp. 58–73, 1981.

[31] D. C. Parkes and Q. Duong, "An ironing-based approach to adaptive online mechanism design in single-valued domains," in *Proc. of the Twenty-Second AAAI Conf. on Artificial Intelligence*, 2007, pp. 94–101.

[32] D. Reiley, S.-M. Li, and R. Lewis, "Northern exposure: A field experiment measuring externalities between search advertisements," in *Proc. of 11th ACM Conference on Electronic Commerce (EC)*, 2010.

[33] A. P. Subramanian, M. Al-Ayyoub, H. Gupta, S. R. Das, and M. M. Buddhikot, "Near optimal dynamic spectrum allocation in cellular networks," in *DySPAN*, 2008.

[34] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *Journal of Finance*, vol. 16, pp. 8–37, 1961.

[35] X. Zhou, S. Gandhi, S. Suri, and H. Zheng, "ebay in the sky: strategy-proof wireless spectrum auctions," in *Proceedings of the 14th Annual International Conference on Mobile Computing and Networking (MOBICOM)*, 2008, pp. 2–13.

[36] X. Zhou and H. Zheng, "TRUST: A general framework for truthful double spectrum auctions," in *INFOCOM 2009. 28th IEEE International Conference on Computer Communications*, 2009, pp. 999–1007.

[37] J. Zhu, B. Metzler, Y. Liu, and X. Guo, "Adaptive CSMA for Scalable Network Capacity in High-Density WLAN: a Hardware Prototyping Approach," in *Infocom*, 2006.

[38] J. Zhu, X. Guo, S. Roy, and K. Papagiannaki, "CSMA Self-Adaptation based on Interferene Differentiation," in *Globecom*, 2007.

Appendix

Proof of Lemma 2

First, we observe that an agent's bid is only used to determine his bucket and is afterward ignored by the algorithm (estimates of utility use the agent's bucket rather than his bid). Thus is it sufficient to consider deviations that cause i to change buckets. If $A_i = \perp$, then $\Pr_i(F|A) = E_A[S_i|F] = 0$, so the claim is trivially true. Otherwise, i moves up to some bucket $k_2 > k_1$. Recall that $\nu_i(A, c) = \{j \in N_i \mid A_j = c\}$ denotes the set of i's neighbors assigned to channel c according to A. An important observation about the algorithm is that once it makes an assignment that some $A_i(k, 0, j) = c$, it never changes this for any later k and j. This is the reason the ironing step only changes *Allocation* and not A. Thus, the set $\nu_i(A(k, 0, j), c)$ grows monotonically as the algorithms iterates over k and j.

Since i was assigned to c in the assignment A, c must have been available to him when he was assigned. By the third part of the definition of availability and the monotonic growth of ν, i would have his demand satisfied with neighbors $\nu_i(A(k, 0, j), c)$ for all $k \geq k_1 + 1$ and all j. In particular, this means his demand is satisfied with neighbors $\nu_0(A(k_2, \pi^{-1}(i) - 1), i, c)$.

When computing *Allocation*$'$ with the new bids b', the algorithm computes a new set of incremental allocations A'. Since the algorithm does not look ahead, $A'(k_2, 0, \pi^{-1}(i) - 1) = A(k_2, 0, \pi^{-1}(i) - 1)$. This means that, in AssignChannel$(A'(k_2, 0, \pi^{-1}(i)), i))$, i could be assigned to c and have his demand satisfied. Therefore he will be assigned to some such channel c' (not necessarily c as there might be a lower numbered channel available). Furthermore, on c he does not impose any externality on his neighbors (all their demands are

satisfied by the second part of the definition of availability). Therefore, since the algorithm greedily maximizes the total value, this true on c' as well.

Again since the algorithm does not look ahead, i increasing his bid does not change anything before bucket k_2, so $A'(k_2+1,0,n) = A(k_2+1,0,n)$. Since the algorithm does not consider allocating i a channel in bucket k_2 when computing A (because he is in the lower bucket k_1) or when asking the counterfactual about what would have happened had i not been in bucket k_2 in A', $A'(k_2,i,n) = A(k_2,0,n)$. Thus $\nu_i(A'(k_2,i,n),c) = \nu_i(A(k_2,0,n),c)$ and in the ironing step running on b'_i,all of i's neighbors with which it might have shared a channel with be reassigned to $\perp$ until its demand is satisfied. Since $i's$ neighbors were satisfied when i was assigned to c' and neighbors are ironed in the opposite order from that in which they were added, i will not be ironed by any of its neighbors. Thus $Pr_i(F|A) = E_{A'}[S_i|F] = 1$ and the allocation is monotone.

Proof of Theorem 3

As observed there are only 3 possible allocations and sets of prices. An agent either gets nothing and pays nothing for a utility of 0, ends up in bucket k in which he might share and gets $v_i a_i fs - p_i a_i(1-f)$ and pays $\beta(k)fs - p_i a_i(1-f)$ for a utility of $(v_i a_i - \beta(k))fs$, or ends up in a higher bucket and gets $v_i a_i$ (he has a channel to himself) and pays $\beta(k+1) - (\beta(k+1) - \beta(k))fs$ for a utility of $v_i a_i - \beta(k+1) + (\beta(k+1) - \beta(k))fs$.

First suppose that $v_i a_i < \beta(k)$. If he ends up sharing his utility is $(v_i a_i - \beta(k))fs < 0$. If he ends up with a channel to himself his utility is

$$v_i a_i - \beta(k+1) + (\beta(k+1) - \beta(k))fs < \tag{0.13}$$

$$(\beta(k+1) - \beta(k))(fs-1) < 0. \tag{0.14}$$

Thus his optimal strategy is to bid his true value and get $\perp$.

Now suppose that $\beta(k) \leq v_i a_i \leq \beta(k+1)$. If he bids truthfully, his utility is $(v_i a_i - \beta(k))fs \geq 0$, so he cannot gain by lowering his bid. If he raises his bid above $\beta(k+1)$ he will end up with a utility of

$$v_i a_i - \beta(k+1) + (\beta(k+1) - \beta(k))fs = \tag{0.15}$$

$$(v_i a_i - \beta(k+1))(1-fs) + (v_i a_i - \beta(k))fs \leq \tag{0.16}$$

$$(v_i a_i - \beta(k))fs. \tag{0.17}$$

Thus his optimal strategy is to bid his true value and share.

Finally, suppose that $v_i a_i > \beta(k+1)$. If he bids truthfully, his utility is $v_i a_i - \beta(k+1) + (\beta(k+1) - \beta(k))fs > 0$, so he does better than if he is not allocated. If he lowers his bid to be in bucket k, his utility is

$$(v_i a_i - \beta(k))fs \leq v_i a_i - \beta(k+1) + (\beta(k+1) - \beta(k))fs \tag{0.18}$$

Thus his optimal strategy is to bid his true value.