Computational Tools for Complex Electronic Auctions

by

Neil Newman

BA.Sc. in Engineering Science, University of Toronto, 2014

M.Sc. in Computer Science, University of British Columbia, 2017

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

The University of British Columbia

(Vancouver)

March 2024

The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, the thesis entitled:

Computational Tools for Complex Electronic Auctions

submitted by **Neil Newman** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy** in **Computer Science**.

Examining Committee:

Kevin Leyton-Brown, Professor, Computer Science, UBC
Supervisor

Jesse Perla, Associate Professor, Vancouver School of Economics, UBC
Supervisory Committee Member

David Poole, Professor, Computer Science, UBC
Supervisory Committee Member

Vitor Farinha Luz, Associate Professor, Vancouver School of Economics, UBC
University Examiner

Bruce Shepherd, Professor, Computer Science, UBC
University Examiner

Sven Seuken, Associate Professor, Department of Informatics, UNIVERSITY OF ZURICH
External Examiner

Abstract

This thesis has two main concerns. The first is infrastructure that allows complex, electronic markets to function, ranging from web applications to highly specialized clearing algorithms. The second is developing computational methods to assess alternative market designs.

I describe efforts developing and deploying computational infrastructure in support of markets in two very different domains: subsistence agriculture and radio spectrum allocation. I detail practical experiences (a) running a feature-phone based marketplace for agricultural trade built to match farmers with traders in developing countries, and (b) designing a solver to overcome the computational challenge of station repacking in the recent US "incentive" spectrum auction.

I then present a series of three computational methods for evaluating alternative market designs, beginning with a setting where plausible models of bidding behavior are known, then relaxing this assumption and studying single-action and later sequential games.

Lay Summary

Markets determine how to allocate resources among self-interested agents. While some markets are simple—e.g., posted prices on products in a grocery store—more complex markets routinely process vast amounts of data and solve challenging optimization problems. While the design and analysis of markets has relied primarily on mathematical analysis, the complexity and scale of many important markets requires augmenting theoretical understanding with computational analysis.

This thesis first describes two market design efforts: an electronic agricultural marketplace in Uganda and a recent US spectrum auction. Second, it introduces three families of computational tools for reasoning about alternative market designs. The key ingredients of these tools are simulations, algorithms to identify computationally useful structure in markets, and computer agents that learn strategies from repeated play. Compelling simulation-based economic analysis requires agents to account for their incentives; our tools leverage machine learning and game theory to identify sets of rational strategies.

Preface

The research presented in this thesis is based on work that was published or is under review.

1. Chapter 2 is based on two publications:

   (a) Designing and Evolving an Electronic Agricultural Marketplace in Uganda. N. Newman, K. Leyton-Brown, N. Immorlica, L. Bergquist, B. Lucier, J. Quinn, C. McIntosh, R. Ssekibuule. ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS), 2018.

   (b) Kudu: An Electronic Agricultural Marketplace in Uganda. N. Newman, N. Immorlica, K. Leyton-Brown, B. Lucier, J. Quinn, R. Ssekibuule. In Artificial Intelligence for Social Impact, M. Tambe, F. Fang, B. Wilder (editors), Springer, 2022.

   I wrote the first draft of the text in both publications, which was later revised by all co-authors. I implemented Kudu's matching procedures (starting from the codebase described in earlier work [Ssekibuule et al., 2013]). I also wrote the code to analyze Kudu's transactions, based on questions arising from discussions with all co-authors. I was the primary developer on the Android application, alongside help from Laura Alvarez and Arman Raina. All co-authors contributed to the market design ideas. Craig and Lauren performed all analysis and design related to the randomized control trials.

2. Chapter 3 is based on the paper

   (a) Deep Optimization for Spectrum Repacking. N. Newman, A. Fréchette,

K. Leyton-Brown. Communications of the ACM (CACM), volume 61, number 1, pp. 97–104 , January 2018.

Alex and Kevin produced the initial version of the feasibility checking tool. I contributed new heuristics to the feasibility checking tool, ran the experiments in the paper, and analyzed the data. Paul Cerneck and Emily Chen also made code contributions to the feasibility checking tool. Kevin and I wrote the text of the paper together. Portions of this chapter are adapted from my MSc thesis entitled "Efficient Feasibility Checking in Reverse Clock Auctions for Radio Spectrum" (2017) completed at the University of British Columbia. I included this work here both to provide context for the following chapter and because it represents a non-trivial example of an auction that could not have been run without modern computational abilities. Pranay Jain helped create the station repacking benchmarks described in this chapter.

3. Chapter 4 is based on the paper

   (a) Incentive Auction Design Alternatives: A Simulation Study. N. Newman, K. Leyton-Brown, P. Milgrom, I. Segal.

   An extended abstract was published at the 2022 ACM Conference on Economics and Computation (ACM-EC). The paper has been accepted for publication in the journal Management Science: Revenue Management and Marketplace Analytics. A modified chapter version of this paper was also published as "Artificial Intelligence and Market Design: Lessons Learned from Radio Spectrum Reallocation. K. Leyton-Brown, P. Milgrom, N. Newman, I. Segal. New Directions in Market Design. 2023."

   I wrote the first draft of the paper, which was later revised by Kevin, Paul, and Ilya. I coded the auction simulator and ran all of the experiments.

4. Chapter 5 is based on the paper

   (a) The Positronic Economist: A Computational System for Analyzing Economic Mechanisms. D. Thompson, N. Newman, K. Leyton-Brown. Conference on Artificial Intelligence (AAAI), 2017.

A first draft of this work appeared as Chapter 7 of David Thompson's PhD thesis "The Positronic Economist: a Computational System for Analyzing Economic Mechanisms" [Thompson, 2015]. I was not involved in this initial draft and took over the project after the draft was written; the main algorithms and theorems in the final paper were already in place. I rewrote approximately 25% of the text, made changes to the black box structure inference algorithm, redesigned and reran all of the experiments, and prepared the work for submission to AAAI.

5. Chapter 6 has been submitted to a conference and is currently under review. The chapter is joint work with Greg d'Eon and Kevin Leyton-Brown. Greg and I both jointly wrote the submission text and supporting code.

Table of Contents

Abstract . . . . . **iii**

Lay Summary . . . . . **iv**

Preface . . . . . **v**

Table of Contents . . . . . **viii**

List of Tables . . . . . **xiii**

List of Figures . . . . . **xiv**

Acknowledgments . . . . . **xx**

Dedication . . . . . **xxi**

1 Introduction **1**

2 Kudu: An Electronic Agricultural Marketplace in Uganda **6**

2.1 Introduction 6

2.2 Problem Statement 9

2.3 Method 10

2.3.1 Gathering Bids and Asks 11

2.3.2 From Proposals to Physical Transactions 16

2.3.3 Matching 21

2.3.4 Facilitating Trade 25

2.3.5 Price Information 27
2.4 Resource Requirements 28
2.5 Field Evaluation 28
2.6 Redesigning the Market To Minimize Human Intervention 32
2.7 Lessons Learned 34
2.8 Conclusions 35

3 Deep Optimization for Spectrum Repacking 36
3.1 Introduction 36
3.2 The Incentive Auction 40
3.2.1 Station Repacking 40
3.2.2 Reverse Auction 42
3.2.3 Worked Example 45
3.3 A Deep Optimization Approach 48
3.3.1 The Design Space 49
3.3.2 Searching the Design Space 53
3.4 Data from Auction Simulations 54
3.5 Runtime Performance 55
3.6 Conclusions 57

4 Incentive Auction Design Alternatives: A Simulation Study 58
4.1 Introduction 58
4.1.1 Evaluating Complex Auction Designs 59
4.1.2 Our Simulation Methodology 62
4.1.3 Questions Considered in Our Analysis 64
4.2 Value and Bidding Models 67
4.2.1 Value Models 67
4.2.2 Bidding Model 69
4.3 Simulator Design and Experimental Considerations 70
4.3.1 Metrics 71
4.3.2 Impairments 72
4.4 Experiments 73
4.4.1 Repacking the VHF Band 73

4.4.2 Multi-Stage Clearing 76
4.4.3 Scoring Rules 82
4.4.4 Feasibility Checking 84
4.4.5 Comparison to VCG in Restricted Setting 89
4.5 Conclusions 91

5 The Positronic Economist: A Computational System for Analyzing Economic Mechanisms 93
5.1 Introduction 94
5.2 Related Work 95
5.3 Mechanisms and Settings 97
5.4 Bayesian Action-Graph Games 97
5.5 Representing Games with PosEc 98
5.6 Structure Inference Algorithms 100
5.6.1 White-Box Structure Inference 100
5.6.2 Black-Box Structure Inference 103
5.7 Experiments and Results 104
5.8 Conclusions and Future Work 108

6 Understanding Iterative Combinatorial Auction Designs via Multi-Agent Reinforcement Learning 110
6.1 Introduction 111
6.2 Methodology: Understanding Iterative Combinatorial Auctions via MARL 115
6.2.1 Modelling an Auction 115
6.2.2 Finding Equilibria 120
6.2.3 Validating and Interpreting Policies 123
6.3 Case Study: Bid Processing in Clock Auctions 125
6.3.1 Clock Auctions 125
6.3.2 Defining the Environment 127
6.3.3 Finding Equilibria 129
6.3.4 Validating and Interpreting Policies 131
6.4 Experimental Results 131

6.4.1 Ablations and Hyperparameter Tuning 132
6.4.2 Effects of Bid Processing Algorithms 134
6.4.3 Scaling to Larger Games 137
6.5 Conclusions and Discussion 138
6.5.1 Promising Auction Design Questions 139
6.5.2 Methodological Improvements 140
6.5.3 Theoretical Questions 141

Bibliography 143

A Supporting Materials: Incentive Auction 162
A.1 FCC Band Plan 162
A.2 Reverse Auction Pseudocode 163
A.2.1 VHF Option Pricing 164
A.3 Additional Details of the BD Value Model 166
A.3.1 Inferring Bounds on Values from Bids 166
A.3.2 Fitting our Value Distribution 167
A.4 Additional Simulation Details 168
A.4.1 Computational Environment 168
A.4.2 Handling Missing Data in the MCS Value Model 169
A.4.3 Additional Value Model Details 169
A.4.4 Initial Band Assignment 169
A.4.5 Canadian Stations 170
A.4.6 Station Volumes 170
A.5 Full Experimental Results 170
A.6 First-to-Finish Algorithm Pseudocode 175
A.7 Impairing Stations 175
A.8 Robustness Experiments 176
A.9 Single-stage vs Multi-stage Auctions Examples 177
A.10 Early Stopping Counterexample 179
A.11 Feasibility Checker Counterexample 180
A.12 Modeling Forward Auction Revenue 180
A.13 VCG MIP Encoding 181

A.14 Station Repacking Benchmarks 182

B Supporting Materials: Reinforcement Learning 184
B.1 Computational Environment 184
B.2 Clock Auctions 184
B.3 Value Sampling Details 186
B.4 PPO Hyperparameters 187
B.5 Additional Results 189

List of Tables

Table A.1 A summary of the results of changing the value model on four replicated experiments from the main paper. Each row corresponds to a different change to the value model, and each column corresponds to an experiment. The "Default" row refers to the original experiment. Values in each cell represent the mean and standard deviation of value loss (upper table) and cost (lower table) of an altered auction design relative to the real auction design across all paired simulations. 178

List of Figures

Figure 2.1 Top Left: Kudu's USSD interface running on a feature phone. Right: Sample USSD interaction for selling groundnuts (peanuts). Bottom Left: A user placing an ask on Kudu's web interface. . 13

Figure 2.2 A histogram of bid prices as a fraction of ask prices and final sale prices for verified transactions. If users were truthful, no trades would ever have occurred with a bid price lower than the ask price; clearly this was not the case. However, most of the mass is distributed around the center, suggesting that user prices still conveyed useful signaling information in most cases. The buyer usually payed slightly less than their bid price in the final sale. 20

Figure 2.3 The manual matching interface on Kudu. Asks are shown on the left and bids on the right. A deal coordinator selected one from each column to create a match. This approach began to break down when the number of bids and asks grew large. . . 23

Figure 2.4 The Kudu AI interface. Deal coordinators were shown matches corresponding to the largest gains from trade. Matches were overlaid on a map of Uganda. Characteristics of the match and the involved users were also shown: for example, we highlighted if the two users have matched together before or are engaged in any ongoing transactions. Deal coordinators could accept or reject proposals, refreshing the available choices. . . 25

Figure 2.5 Active users over time. Each bar represents the number of unique active users in a one month interval. New users that month are highlighted. An active user is defined as a user who used any of Kudu's services during the given month. In total, 11,861 unique users engaged with the platform. 29
Figure 2.6 Cumulative value of verified transactions between September 2016 and March 2018. 29
Figure 2.7 Geographical range of our verified transactions, with edges linking the reported parishes (small villages) of buyers and sellers respectively. About a third of verified transactions occurred within the same parish and are not visible on this map. The large clusters correspond to the 11 districts throughout Uganda in which Kudu was supported through in-village services. We note that Kudu spread beyond these treatment districts. 30
Figure 2.8 Screenshots of the Kudu application running on an Android smartphone. Users select from a menu of crops and can filter on features of the listing such as price, quantity, recency, and geography. Users interested in a listing receive the owner's contact deals and manually initiate a trade. 32

Figure 3.1 Interference graph derived from the FCC's constraint data. . . 42
Figure 3.2 An illustration of the reverse auction example described in Section 3.2.3. Each row corresponds to a stage, and a new figure is drawn each time a station exits. Active stations are dashed, exited stations are solid, frozen stations are dotted, and stations with catch-up status are dashdotted. Connected stations cannot jointly broadcast on the same channel. B and C additionally cannot jointly broadcast on adjacent channels (shown by the thick bold edge between them). To the left of each figure is the current clock price. 46

Figure 3.3 Empirical Cumulative Density Function (ECDF) of runtimes for default configurations of MIP and SAT solvers and for SATFC 2.3.1. The curves show fraction of instances solved (y axis) within different amounts of time (x axis; note the log scale). The legend is ordered by percentage of problems solved before the cutoff. The histogram indicates density of SAT and UNSAT instances binned by their (fastest) runtimes; unsatisfiable instances constituted fewer than 1% of solved instances. . 56

Figure 4.1 An overview of our simulation approach. N samples from a bidder model are fed into both a modified and unmodified simulator. The two outcome sets are converted into metrics and compared on a per-sample basis. 63
Figure 4.2 CDF for our maximum likelihood estimate of N and its log-uniform distribution fit, plus generalized Pareto left and right tails. 68
Figure 4.3 Comparing auctions that only repack the UHF band against auctions that also repack VHF bands. 74
Figure 4.4 Comparing auctions running through 1-4 stages, ultimately ending on the same clearing target, with no impairing stations. . . 75
Figure 4.5 Comparing early stopping auctions against single-stage auctions. 80
Figure 4.6 Comparing auctions using four different scoring rules. 80
Figure 4.7 Comparing auctions using different feasibility checkers. . . . 85
Figure 4.8 Comparing the first to finish algorithm against the standard bid processing algorithm for single-stage 126 MHz auctions. . . . 86
Figure 4.9 Interference graph of the set of 218 UHF stations within two edges of a New York station under a 126 MHz clearing target. 91
Figure 4.10 Comparing VCG auctions and reverse auctions in Greater New York City simulations. 92

Figure 5.1 Sample code defining a mechanism and setting for plurality voting with randomized tie breaking. 99

Figure 5.2 WBSI's first steps on the plurality voting example in Figure 5.1. Beginning with a disconnected action graph (a), WBSI selects an action node (here, voting for candidate A with type $A \succ B \succ C$) and creates an empty payoff table. As the choice function is run, WBSI will infer from accessor calls such as `a_N.count("A")` that it needs to create summation nodes aggregating the number of players that voted for each candidate across types, and that these function nodes should be inputs to the payoff table. (b) shows the action graph once the payoff table is computed, and (c) shows the payoff table (for a 2 agent game). 101

Figure 5.3 Encoding sizes (first row); Encoding times (second row); Median time required to identify an equilibrium using the GNM algorithm (third row), IBR algorithm (fourth row), and a parallel portfolio of GNM and IBR running on two cores (fifth row). WBSI+BBSI achieved substantially more compression than WBSI only in the case of wGSP; this suggests that previous work identified very effective encodings. We did not run BBSI on games with more than 5 players as the runtime required grew prohibitively large. All equilibrium finding plots are truncated at our budget of one hour; following the *penalized average runtime* (PAR10) metric, timeouts are logged as ten hours. 105

Figure 5.4 Median regret (across random initializations) achieved by IBR in (a) and FP in (b)–(c) on 10 agent games as a function of time; each line corresponds to a different game and regret is normalized by the maximum payoff in each game. 106

Figure 6.1 The AuctionNet neural network architecture. 130

Figure 6.2 NashConv of MCCFR ablations, varying secondary rewards and trembling opponents. 133

Figure 6.3 NashConv of PPO hyperparameter tuning runs, showing all 60 configurations (left) and a detailed view of the 20 best configurations (right). Configurations are sorted by 95th percentile of NashConv. Whiskers denote 5th and 95th percentiles. . . . 133
Figure 6.4 Auction outcomes using MCCFR and PPO on 2-player games with 1 to 7 types. 135
Figure 6.5 Auction outcomes under straightforward bidding on 2-player games with 7 types. 136
Figure 6.6 (a) NashConv runtime distribution; (b) auction outcomes using MCCFR on 3-player games. 138

Figure A.1 The FCC's band plan for 600 MHz. Each row corresponds to a different clearing target and shows which channels would be repurposed and which would remain for TV broadcasting. . . 162
Figure A.2 Comparing auctions that only repack the UHF band against auctions that also repack VHF bands. 171
Figure A.3 Comparing auctions running through 1-4 stages, ultimately ending on the same clearing target, with no impairing stations. . . 171
Figure A.4 Comparing auctions running through 1-4 stages, ultimately ending on the same clearing target, factoring in impairing stations. The BD value model is used for all simulations. 172
Figure A.5 Comparing auctions using the early stopping algorithm against single-stage auctions ending on the same clearing target as their corresponding early stopping auction. 172
Figure A.6 Comparing auctions using four different scoring rules. 173
Figure A.7 Comparing auctions using different feasibility checkers. . . . 173
Figure A.8 Comparing the standard bid processing algorithm with one that does not freeze stations with indeterminate feasibility checks. 174
Figure A.9 Comparing the first to finish algorithm against the standard bid processing algorithm for single-stage 126 MHz auctions. . . . 175
Figure A.10 Returning to the example of Figure 3.2, but now repacking two channels right from the outset. 178

Figure A.11 Forward auction revenues observed in the incentive auction ("Actual") and our model of forward auction revenue. 183

Figure B.1 An example value profile for a game with two bidders and four types. Fractional numbers of licenses on the x-axis correspond to bundles with encumbered licenses. 187

Figure B.2 Auction outcomes using MCCFR on 2-player games with 1 to 7 types. 190

Figure B.3 Auction outcomes using PPO on 2-player games with 1 to 7 types. 191

Figure B.4 Auction outcomes under straightforward bidding on 2-player games with 1 to 7 types. 192

Acknowledgments

I would like to acknowledge:

- my supervisor, Kevin Leyton-Brown, for all of his guidance and advice.
- Jesse Perla and David Poole, my supervisory committee.
- Sven Seuken, my external examiner.
- Kevin Song, for chairing my examination.
- the Auctionomics team, for providing me with practical auction experience that informed much of this work.
- Marc Lanctot, for providing advice on reinforcement learning.
- undergraduates I've had the pleasure of working with: Paul Cernek, Emily Chen, Peter West, Arman Raina, and Laura Alvarez
- Alexandre Fréchette, who helped me settle into graduate school by providing mentoring and guidance, and for laying the groundwork for this research direction through his initial work on the incentive auction.
- the GTDT research group spanning all of my years here: Jason, Chris, Alice, Lars, James, Hedayat, Devon, Taylor, Greg, and Narun. It would have been much a duller degree without each of you!
- the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Digital Research Alliance of Canada for funding and computational resources.

For Mom, Dad, and Robert. Thanks for all your support.

Chapter 1

Introduction

Markets determine how to allocate resources among self-interested agents. While some markets are simple—e.g., posted prices on products in a grocery store—modern markets routinely process vast amounts of data and solve challenging optimization problems. As a result, market design has shifted from pure economics to a partnership also including operations research and computer science. These collaborations span a bewildering number of settings, including domains that may not traditionally be thought of as markets. Some examples include allocating licenses to radio waves [Leyton-Brown et al., 2017]; allocating fishing licenses [Bichler et al., 2019]; allocating and rebalancing docks in bike sharing networks [Chung et al., 2018, Freund et al., 2018]; resettling refugees [Delacrétaz et al., 2023]; selecting school assignment policies [Allman et al., 2022]; determining patrolling strategies for wildlife protection [Yang et al., 2014], and even pairing organ donors and recipients [Abraham et al., 2007, Santos et al., 2017].

In this interdisciplinary cross-pollination, computer science offers analytic tools such as algorithm analysis and complexity theory. Yet market design is as much an engineering discipline as a science. As noted by Al Roth in his article *The Economist as Engineer*: "Market design involves a responsibility for detail, a need to deal with all of a market's complications, not just its principle features..." [Roth, 2002]. These "complications" can violate assumptions in models, which make simplifications out of necessity, even though it is not usually a priori obvious which dynamics will be important to retain.

We routinely deploy hard-to-reason-about markets for which practice has outpaced theory. Fortunately, complementary to its analytic toolbox, computer science also offers an empirical toolbox, continuously bolstered by Moore's law. Within it lie tools such as heuristic algorithms, data science, machine learning, and Monte Carlo simulations. When used prudently, these tools can augment theoretical insight and offer a way to analyze to otherwise intractable markets.

This thesis has two main concerns. The first is infrastructure that allows complex, electronic markets to function, ranging from web applications to highly specialized clearing algorithms. The second is developing computational methods to assess alternative market designs through simulations. As such, the first two chapters of this thesis describe our experiences developing and deploying computational infrastructure in support of markets in two very different domains, while the latter chapters shift to computational market analysis, beginning with a setting where plausible models of bidding behavior are known, then relaxing this assumption and studying single action and later sequential games.

This thesis includes practical applications to two market settings: electronic agricultural markets in developing countries, and auctions for radio waves (spectrum). These two settings differ in significant ways. The agricultural setting is characterized by technologically unsophisticated users and relatively straightforward interactions. In contrast, spectrum auctions are infrequent, lengthy processes involving highly strategic bidders and extremely complex rules. Another key difference is that while farmers and traders are faced with multiple markets for buying and selling crops, telecom companies can only obtain spectrum through centralized government auctions. Despite their apparent dissimilarities, both markets are ripe candidates for the computational toolbox: they are informed by microeconomic theory, but grounded in messy, real-world constraints that make theory challenging to directly apply.

The rest of the thesis proceeds as follows. First, Chapter 2 describes our experiences running Kudu. Kudu was an agricultural market we operated in Uganda, most active between 2015–2018. The market's goal was to alleviate the difficulties that smallholder farmers face in finding counterparts for trade. Farmers and traders sent information to a centralized database about what crops they wanted to buy or sell, and Kudu suggested profitable pairings of users. A key technological con-

straint was that most users could communicate only through platform-subsidized SMS or phone calls; they did not have internet access or smartphones. Despite initial traction, the market relied substantially on paid labor to operate and was shut down due to a lack of funding and no clear path towards profitability. The computational infrastructure powering this project included matching algorithms for pairing users, as well as SMS, USSD, web, and smartphone interfaces that allow users to effectively communicate their preferences within the technological limitations of each medium.[1] Subsequent studies by collaborating development economists concluded that Kudu represented a net welfare gain, with improvements to farmer welfare exceeding the operating costs and harm to traders' profits.

Next, Chapter 3 covers the "incentive auction", a 2016–2017 US auction that reallocated spectrum from television stations to mobile carriers. The auction format was novel and a prime example of computer science and economics intertwined in market design: it required solving tens of thousands of NP-complete problems under tight time constraints. These problems resemble graph colouring problems[2] with television stations as nodes and television channels as colors; edges are based on interference constraints due to physical properties of the stations' towers. Computational tractability was a first-order concern. We leveraged tools from the study of heuristic algorithms to design the *feasibility checker* used in this auction to solve these problems.

Auction rules tend to evolve over time: the initial spectrum auction formats of the 1990s look quite different than those of today, partially as a response to undesired bidder behavior and partially because increasing computational power has made new designs viable. How the outcomes of an auction, such as revenue, might change as rules are tweaked is not typically obvious as bidders may vary their strategies. We develop methods an auctioneer can consult to forecast likely impacts of counterfactual design choices. Our methods are particularly suited to cases where complexity precludes analytical results and a lack of data precludes structural estimation.

[1] Interfaces to markets really matter: Milgrom [2004] recounts that early FCC software did not allow commas in the bid field; on several occasions bidders made "fat finger bids" that were orders of magnitude different from what they intended.

[2] The problems generalize graph colouring problems, since interference can bleed across neighbouring channels. For more details, see Section 3.2.1.

After the incentive auction concluded, we performed a retrospective analysis of several design choices, with the goal of identifying which features were most important, and which variations of the design might have led to even better outcomes. We focused specifically on the novel *reverse auction* half of the incentive auction, used to procure stations' broadcast rights. Computational simulations were a natural fit for this analysis: the interference constraints are very heterogeneous across stations, making the auction difficult to reason about analytically. We built a reverse auction simulator and developed a realistic model of bidder valuations based on publicly released bid data. Chapter 4 details how we compared various design alternatives based on the outcomes of many simulations. This approach—running Monte Carlo simulations to examine economic outcomes under a large space of design choices and bidder behaviors—is generally applicable to novel markets that have been proposed in the literature but have not yet been implemented in practice or those with limited historical data.

The main barrier to applying simulations for market counterfactuals more broadly is that they require a compelling model of how agents bid. The most prevalent approach in the literature is to claim that bids will constitute (some flavor of) "equilibrium". For example, under a Nash equilibrium no agent would want to unilaterally change their strategy given knowledge of others' bids. When agents have a *dominant* strategy (and one expects that agents are aware of this strategy)[3], it is sufficient to sample auction outcomes, varying valuations and randomness in the auction itself. When an equilibrum strategy is not known, it must be solved for either analytically or computationally. Finding a set of equilibrium strategies for large games like spectrum auctions is very computationally challenging. Formally, the problem is either NP-complete or PPAD-complete depending on the particular equilibrium variant being identified; in practice, this means that running times are at least exponential in the description size of the game.

Chapter 5 introduces a tool, The Positronic Economist, which automatically infers *compact representations* of one-shot, simultaneous move games. When games

[3]It can really matter that a strategy's dominance is "obvious" to human players: e.g., Gonczarowski et al. [2022] show that humans behave differently based on the exact phrasing used to describe a mechanism in which they can do no better than simply reporting their preferences truthfully.

possess the right structure, these representations can exponentially speed up equilibrium finding. Compact representations are laborious to design and are specific to individual mechanisms. Our software package automates the design of compact mechanisms by inferring useful structure from a Python-based description of a mechanism.

The Positronic Economist cannot be applied to dynamic games, such as iterative combinatorial auctions. In Chapter 6 we ask whether this hole can be plugged by modern reinforcement learning algorithms. We show that, with careful modeling choices, multi-agent reinforcement learning can identify near-equilibrium (and often exact equilibrium) strategies in modestly sized auctions. Then, focusing specifically on clock auctions, we demonstrate how an auction designer can study how these equilibria shift outcomes under a suite of rule modifications to inform design.

Chapter 2

Kudu: An Electronic Agricultural Marketplace in Uganda

This chapter describes our experiences designing and operating an electronic market platform for agricultural trade, branded in Uganda as *Kudu* [Newman et al., 2018]. We examine several design choices related to the computational infrastructure we deployed, including both the interfaces we developed for end-users and our staff, as well as the evolution of our market clearing algorithm. Complexity in this setting stems from the many challenges of operating in a developing country, including costly communication and not having the ability to enforce commitments.

2.1 Introduction

A significant challenge facing rural development is inefficiency in agricultural markets. One potential driver of such inefficiency is farmers lacking information about the national market for their crops and therefore selling in local markets at suboptimal prices. The result is not only lower prices for farmers (often a huge group, since 80% or more of the population in many African countries work in agriculture [Farm Africa, 2018]), but also intra-seasonal and cross-locational price fluctuations that distort the market and reduce incentives for investing in productivity-enhancing

inputs. Prior work [Ssekibuule et al., 2013] has demonstrated the existence of arbitrage opportunities both via buying and selling in different parts of the country as well as via paying for crop storage between seasons. Such inefficiencies are driven by information failures: market discovery occurs almost entirely through word-of-mouth interactions; buyers and sellers settle on prices through negotiation. Most gains from trade are captured by better-informed intermediaries [Bergquist, 2017]. Worse still, when both parties are insufficiently well informed, mutually beneficial trades simply may not occur [Aker, 2010, Jensen, 2007]. In the long run, without accurate knowledge of nationwide agricultural demand, it is difficult for farmers to make good decisions about which crops to plant.

The internet has revolutionized many two-sided markets by making it easy for market participants to discover current conditions and to find each other. We were motivated by the idea that if farmers were both better informed about market conditions and better empowered to reach out to buyers beyond their immediate social networks, they would achieve better market outcomes. Unfortunately, there was a massive hurdle to setting up an electronic marketplace in small-scale Ugandan agriculture: our potential user base consisted of smallholder farmers—farmers growing mainly for subsistence who occasionally have crops to sell—who have limited or no access to the web. However, penetration of feature phones—phones capable of sending and receiving voice calls and SMS messages, and running USSD applications—was nevertheless high. For example, the World Bank estimated that there were 55 mobile subscriptions per 100 people in Uganda in 2016 [Bank, 2018]. We therefore set out in 2011 to design an electronic marketplace in which a user could fully participate using only a feature phone.

In brief, our system operated as follows. Farmers and traders used their mobile devices to place *bids* (requests to buy) and *asks* (requests to sell) into a centralized nationwide database. Kudu identified profitable trades, which were then proposed to the corresponding participants. Users' trust in the system was enhanced by the availability of in-village support services, provided by Agrinet, a private-sector Ugandan intermediary; users were supported by a call center. Our platform also gathered price data and broadcasted it back to farmers and traders using SMS, drawing from a large set of national, regional, and local markets and providing a uniquely tailored information set to each user.

Kudu was first piloted in 2013 [Ssekibuule et al., 2013]; after a brief hibernation, it rebooted in partnership with Agrinet and Innovations for Poverty Action in May 2015. In this iteration, Kudu was part of a multi-year randomized control trial to assess its role on farmer welfare. While anyone was free to register and use Kudu, we only advertised and offered in-village support in certain parts of the country. We did not charge users anything to use Kudu; instead, a combination of grants and self-funding covered Kudu's expenses (which were dominated by the cost of human employees). The marketplace was then active for nearly three years and registered over 21,000 users through radio ads, village promotion meetings, and word of mouth. Users submitted nearly 30,000 asks and over 30,000 bids, resulting in more than 850 verified completed transactions involving over 5,000 tons of grain with a value of more than $1.9 million USD[1,2]. Grant funding for the project concluded in March 2018 and the market has been largely inactive since this time; efforts to commercialize operations were non-trivial to institute in our setting (see Section 2.6 for more details). The headline result from the randomized control trial was that Kudu had a modest impact on reducing price dispersion between nearby markets. A back-of-the-envelope calculation by our collaborators [Bergquist et al., 2021] suggests that Kudu may have yielded net positive welfare benefits on the order of over $30M (after taking into account a projected reduction in trader profits of $1.53M and the $1.39M cost of running the intervention). While the error bars on this calculation are large and so the raw numbers should be taken with a grain of salt, the key observation is that so many farmers were affected by Kudu that the system would have yielded positive overall welfare effects as long as the average farmer benefited even slightly. We discuss these results in more detail in Section 2.5 .

[1]There are some (relatively small) discrepancies between the numbers reported above and those reported by our collaborators in Bergquist et al. [2021]. Substantively there are two main causes: first, a subset of market activity from users belonging to Agrinet was not posted directly to Kudu and is therefore excluded from our metrics. Second, we differed in whether we recorded users who asked to place recurring "persistent" bids as having placed a single bid or as having placed multiple separate bids.

[2]For comparison, Uganda's agricultural sector is responsible for 24.5% of GDP, having a total value of about $6.5 billion USD.

2.2 Problem Statement

The problem motivating our work is that agricultural markets in developing countries such as Uganada have very high search costs, leading to inefficiencies. After factoring in transportation costs, there can be significant variation in commodity prices between locations in violation of the "law of one price". Using self-reported transportation costs from traders to construct a per-kg estimate of the cost of moving crops, our collaborators found differing prices between nearby markets that could not be explained by the cost of transportation alone [Bergquist et al., 2021]; we reached a similar conclusion in an earlier study [Ssekibuule et al., 2013]. Our hypothesis was that an electronic marketplace could empower smallholder farmers by connecting them to new trading partners and improving their awareness of prevailing prices. Our main design constraint was that the market had to be entirely operable using a feature phone.

There have been attempts in the past to improve agricultural markets through price advisory systems. Examples include Esoko's commodity index [David-West, 2010], Farmgain Africa [Farmgain Africa, 2018], and Infotrade Market Information Services [Infotrade, 2018].[3] These services typically offer SMS subscriptions and radio based market information. However, experimental evidence that price advisory systems have been effective in improving farmer welfare is mixed [Camacho and Conover, 2011, Fafchamps and Minten, 2012, Hildebrandt et al., 2020, Mitra et al., 2018, Nakasone, 2014]. These systems are typically based on manually gathered quotes that are sparse, geographically coarse, and biased by participants seeking to skew the reported statistics. Also, these systems often report only a single number (e.g., mean price) rather than distributional information; distributional information can inform farmers about how to price their crop based on how urgently they need to sell. Moreover, evidence suggests that simply providing price information may be insufficient for farmers who do not have the means of actually accessing the better markets about which they may learn [Mitra et al., 2018]. Smallholder farmers may need connections to specific buyers in these new markets or, in the likely event that they lack the ability to transport their crops themselves,

[3] See Section 1.2 of Ssekibuule [2017] for a survey of agricultural price information systems in Uganda.

they may even need those buyers to come to them. Kudu was aimed at comprehensively addressing this set of barriers to market access. It went beyond previous services, offering nuanced market information, direct market connections, and wraparound services needed to provide smallholder farmers truly improved market access.

A market designer needs to do more than just provide a means for people to interact with the market: they must encourage participation by making the market simple to use and its benefits obvious, ensure that strategic gaming does not undermine the market, and make certain that even as the market grows, finding a trading partner does not become overwhelming. Solutions to these challenges take different forms in different marketplaces: see Kominers et al. [2017] or Einav et al. [2016] for surveys of how marketplaces tailor solutions according to their unique constraints. In addition to the aforementioned technological hurdles, unique challenges in our setting include technically unsophisticated and even illiterate users, the need to limit communication due to airtime costs, cultural resistance to adopting electronic markets, and high travel costs.

One alternative system design—which we rejected very early on—would simply have offered a database of bids and asks that users would have had to search manually. We rejected this idea because we believed that solely offering self-serve ads would not be enough to instill credibility and because we did not believe that searching through listings could be made effective on feature phones, especially when a prime consideration was location.

Artificial intelligence has been recognized as playing an increasingly important role in market design [Milgrom and Tadelis, 2018], for example to reduce search frictions. As we will describe later in Section 2.3.3, the key AI problem in Kudu was to decide what matches to propose to users and when to propose them. This involved both selecting the proper matching algorithm and accurately predicting whether proposed trades would be successful.

2.3 Method

In this section, we provide a detailed breakdown of how Kudu operated and some of the major challenges we faced. We begin with how Kudu gathered bids and asks

(Section 2.3.1), including some reflections on pain points of various interfaces. Then, we discuss how matches flowed through our system and reasons why our proposed matches often failed to become deals (Section 2.3.2). Next, we explain the evolution of our matching procedure in Section 2.3.3. Finally, we describe additional support we offered to facilitate trade (Section 2.3.4).

2.3.1 Gathering Bids and Asks

To place bids (asks) on Kudu, a user needed to tell us what crop they wanted to buy (sell), their requested buy (sell) prices, and desired (available) quantities[4]. Our services were available in four languages: English, Luganda, Luo, and Runyakitara. Our marketplace supported 76 crop types.[5] Crops differ in quality. This was problematic, because we wanted Kudu to be able to treat competing asks as interchangeable. After much reflection and user feedback, we did not adopt a quality grading system; two key hurdles were enforcement and inconsistency in users' abilities to grade crops effectively. Instead, we solicited bids and asks in terms of "fair average quality," inviting traders to negotiate a price adjustment at transaction time to deal with deviations from this quality level. Despite its inelegance, this system worked well in the Ugandan cultural context where point-of-sale bargaining is already common; this design choice did not lead to significant pushback from users.

When a new user placed a bid or ask on our system, we "registered" the user. One key fact we recorded about each user was their location, stored at the *parish* level (an administrative unit in Uganda made up of a small number of villages; Uganda has about 5,000 parishes). Parishes are grouped together into (nearly 1,000) *subcounties*, which form 136 *districts*, which in turn combine into 4 *regions*. Our assumption that people occupy fixed (and arbitrary) locations within a parish is obviously a coarse one; however, in a survey of our users, we determined that this assumption was reasonable for about 85% of them, and hence decided that a more complex system would not justify its cost.

[4]The units in which quantities were specified depended on the crop and reflects how they would usually be advertised. For example, bids for maize specified the desired weight in kilograms, whereas bids for potatoes specified the desired number of sacks.

[5]We refer to anything sold on Kudu as a crop, but a small fraction of our supported commodities were not plants, such as eggs, fish, and livestock.

Over the course of the project, we experimented with ways of enhancing the bidding language: for example, at one point we allowed "location filters" that specified that a buyer would only consider traveling within a specific geographic region. We dropped this feature because it was not well utilized; instead, we eventually accounted for travel costs when proposing matches (see Section 2.3.2).

To avoid hassling potential users with a complicated authentication system, we did not require users to set a username and password. Instead, users on Kudu were identified by their phone numbers. We worried about this breaking down when a user changed their phone number, or when multiple people with different devices wanted to share a single account, but it worked well in the common case.

There were four ways that users could interact with Kudu: sending an SMS, using our USSD application, visiting our website, or speaking to our call center. Each of these interfaces could be used to buy, sell, or request price information.

SMS

Users could send a toll-free SMS to our short code using any of the following templates:

buy [crop] [quantity] [unit price]
sell [crop] [quantity] [unit price]
price [crop]

If a user entered a crop name that did not exactly match one known in the system, we searched for a close match and automatically corrected it. The user received a confirmation SMS with their bid or ask and could call us if anything was wrong.

With the SMS interface, we were often unable to automatically parse messages because they strayed too far from the template. We received more than 3,000 SMS messages that could not be parsed. We failed to parse messages for many reasons: some had nothing to do with agriculture (e.g., questions about voting and health), some were written in an unsupported language, some were missing required information (e.g., the SMS consists of the single word "buy" or "sell"), and some

Kudu

Sell Produce

Highlighted fields are Required

Produce

Maize

Quantity

1000

Price (UGX)

800

Sell produce

Help On Selling »

Produce You're Selling »

(00:47:58) Kudu:
Welcome to Kudu! Please reply with:
1. Buy
2. Sell
3. Price Information
4. Block Farmer / Trader
5. Language / Orulimi / Leb

(00:48:16) User: 2 (Sell)

(00:48:16) Kudu:
What crop would you like to sell?
1. Maize
2. Nambale beans
3. Black beans
4. Mixed beans
5. Other
0. Back

(00:48:25) User: 5 (Other)

(00:48:25) Kudu:
Enter crop
0. Back

(00:48:39) User: GNUTS

(00:48:39) Kudu:
Select option below or enter 0 and try again
1. Groundnuts
2. Goats
3. Piglets
4. Eggplant
0. Back

(00:48:46) User: 1 (Groundnuts)

(00:48:46) Kudu:
Enter quantity in Kgs
0. Back

(00:48:58) User: 700

(00:48:58) Kudu:
Enter unit price (per Kgs)
0. Back

(00:49:07) User: 4200

(00:49:07) Kudu:
Sell 700 Kgs of Groundnuts for 4200/= per Kgs?
1. Submit
2. Main menu
0. Back

(00:49:17) User: 1 (Submit)

(00:49:17) Kudu:
Your sell of 700 Kgs of Groundnuts for 4200/= per Kgs has been placed on the market.
You can call 0780997402 for help.

Figure 2.1: Top Left: Kudu's USSD interface running on a feature phone. Right: Sample USSD interaction for selling groundnuts (peanuts). Bottom Left: A user placing an ask on Kudu's web interface.

mirrored the template (i.e., literal texts of "buy crop quantity price"). Other common mistakes were including units or descriptive information ("SELL DRYED CASSAVA 5000KGS 1500 PER KG"), or treating the SMS like a newspaper ad ("BUY GINGERS, LOCAL-400000SHS,FOREIGN-300000SHS,PER SACK OR 120KGS, ANY N0. OF SACKS OR KGS.CALL 256*********"). We assembled all of the messages that could not be parsed and our call center staff corrected these messages as they were able, phoning users when necessary.

Even when an SMS matched a template exactly, our system could still fail to capture the user's intent. If the crop name was misspelled, for example, our system could make the wrong correction. Users could also accidentally reverse the ordering of the positional quantity and price arguments, and both numbers could sometimes be in the same ranges making this difficult to identify (perhaps advocating for named arguments).

One of the main disadvantages of SMS is that it is not intuitive and requires training. An initial trial found that it was too difficult to register users via SMS (we requested that users send a "parish [parish]" message, but few did and it was hard to disambiguate between similar sounding parishes). Ultimately, first-time SMS users received a phone call from our call center to confirm this information. This and other trials have taught us that our SMS templates were not very flexible, limiting our ability to make changes to the bidding language over time.

We found that many users were able to grasp the SMS format after training, and the SMS system was inexpensive to run. However, our expectation was that it would be made obsolete by the presence of USSD, described below. In a six month study between September 2017–February 2018, SMS accounted for only 0.17% of bids (13) and 1% of asks (72).

USSD

Unstructured Supplementary Service Data (USSD) allows the user of a feature phone to open a real-time connection to an application and to engage in two-way data exchange, creating a responsive experience. A familiar example is an application for purchasing airtime. See Figure 2.1 for a sample Kudu interaction.

USSD solved many of the issues with SMS: a user could learn to navigate the

interface independently; bids could be previewed before submission (allowing a user to confirm that the information was accurate); error messages could be reported in response to nonsensical inputs (e.g., 0 quantity). All of this was in principle possible with SMS but would have been unwieldy, requiring multiple back and forth messages. USSD has further advantages that are not implementable via SMS: e.g., one can implement a password login; sensitive messages are not stored on the device. We also found that having a USSD application was a sign of prestige, and in addition to the advantages we have described it acted as a strong signal to users that our service was backed by a serious enterprise. For all of these reasons, USSD has also been used in other development projects [Perrier et al., 2015].

Unfortunately, despite all of the positives just discussed, USSD came with its own set of issues. One key problem was that it did not allow messages longer than 182 characters. This was very restrictive in practice: it made selecting from long lists difficult, such as when disambiguating parishes with similar names. Furthermore, and most importantly, sessions longer than 2 minutes timed out, leaving the user to start from scratch. This could lead to very frustrating experiences when menu sequences were long and when users had not prepared answers to all of the questions in advance. First-time USSD users trying to buy or sell were prompted for additional information to register, further exacerbating the time limit issue. In the end, we still had to dedicate call center employees to identifying incomplete USSD sessions and calling users back to place their bids for them.

We launched our toll-free USSD application in November 2015. Most USSD usage was to check price information, but in the six month study period mentioned above it also produced 1.2% (74) of our bids and 5.4% (383) of our asks. Our USSD service went offline on January 31, 2018 because our provider unexpectedly shut down all USSD services. We never revived the USSD application.

Web

We provided a web interface to Kudu as shown in Figure 2.1. While we did not expect this option to be used by smallholder farmers, the web interface was important for discovery and may have been appealing for more technologically sophisticated users. In the six month study period, 0.6% (35) of all bids and 0.6% (42) of all asks

were placed via the web interface.

Call Center

Ultimately, as the observant reader will have noticed, the vast majority of our bids and asks in our current system did not come from any of our three "self-serve" options. Instead, in our six-month period of investigation, 97.7% (6167) of bids and 91.6% (6471) of asks were solicited by our call center. Monday through Friday, agents called traders and farmers and asked them if they had anything to buy or sell. This was very effective at thickening the market, and did not require users to be technologically sophisticated. However, it was very labor intensive—scaling linearly in the number of users on Kudu—so was not an economical approach in the long term.

Lastly, we note that a small fraction of bids and asks also came into our system by users calling in. We suspect that some users found this approach more convenient than dealing with our other interfaces.

The above described ways in which users contributed information into the system. We also sometimes needed to contact users at other times, for which we relied on phone calls and SMS messages. Calling a user is flexible and eliminates ambiguity about their intentions, but was expensive and required reaching the user over the phone. SMS messages were cost effective and could be managed on the user's own time but were problematic when users were unresponsive, e.g., due to illiteracy, lack of battery power, or sharing a phone among multiple family members, and required significantly more trust in the electronic matching system. We were strongly considering augmenting phone calls with an Interactive Voice Response (IVR) system to help illiterate users if the project had continued.

2.3.2 From Proposals to Physical Transactions

Once Kudu gathered bids and asks, the next step was to make matches. We will discuss the matching process in more detail in the following section; this section will focus on what happened after we proposed a match. During the 2013 pilot, when we matched two users, we sent them a text with each other's phone numbers, wished them a successful matching, and left them to their own devices. For some,

this was enough to spark a transaction, but for many an automated text telling them to contact a random stranger did not instill enough confidence to lead to a trade. In the second wave, we employed *deal coordinators* to shepherd matches along and introduce a human element into the system. By the time phone numbers were exchanged, the deal coordinators had already spoken to the seller and buyer individually and could vouch for one to the other, incorporating past experiences when available.

When a match was proposed, a deal coordinator first called the seller. If the seller was interested in the match, they next called the buyer. Pending buyer interest, phone numbers were exchanged. The deal coordinator checked in with both parties to see if an agreement was reached. Finally, they followed up after the deal date and recorded what transpired. Of course, either side could pull out at any step in this process.

An additional job for deal coordinators was to look at asks that they were unable to match at the end of every day and give feedback about why they did not match (e.g., price or quantity too low) and allow users to change their prices.

Why Proposals Failed

We proposed trades that we genuinely thought would be profitable, yet only a small fraction of the trades that we proposed actually occured: e.g., in a detailed investigation between September 2017–February 2018, only 7% of the asks that we proposed matches for led to deals. While we certainly expected some failures, given that these were trades that Kudu identified as "profitable", it is worth asking why the fraction of failures was so large. The first, and by far most common, reason is that by the time we proposed a match, at least one of the parties was no longer in a position to trade. The second was that Kudu's assessment of what trades were profitable (based on the existence of a bid–ask spread) did not necessarily correspond to users' own assessments. Sometimes such mismatches were because of price or quantity issues, but most commonly the problems arose because of travel costs.

Transacting Outside The System Farmers are highly liquidity constrained. They tend to sell crops when they urgently need money, e.g., for school fees or medical

bills. Cash kept on hand can be stolen or preyed upon by extended family and friends looking for hard-to-refuse loans. Therefore, when a user notified us that they wanted to sell, we needed to move quickly; a common failure mode of our proposals was that the seller had already sold their crop by the time we sent them a match. Conditioning on only those trades where the seller was still interested in transacting upon being contacted with a match, our success rate jumped to 16% (i.e., more than double). Understanding the need to sell quickly prompted us to institute expiry dates for bids and asks (7 and 3 days respectively). After this duration, unless we heard otherwise from the user, we assumed that a bid or ask was no longer valid.

Even if we did propose a match in time, and negotiations were successful, transacting outside of the system was still a concern. When a buyer traveled to meet their matched farmer, on the way they could encounter another farmer selling exactly what they wanted. If this occurred, they sometimes did not feel obligated to continue onwards to transact with the intended recipient, and instead took the closer trade. The only defense we had against this behavior was a reluctance among buyers to jeopardize their reputations with our deal coordinators.

Geographic Constraints After discounting trades that failed because one party had already sold, the next most common failures had to do with one party not wanting to travel. In order to accurately estimate trade profitability, it was important to develop a model of transport costs. We had coordinates for the location of every parish. One might guess that transportation costs would have been roughly linear in Euclidean distance (we initially did), but this turned out to be highly inaccurate because of bodies of water, mountainous regions, and road quality issues. It was also tempting to use Google Maps to estimate travel times, but unfortunately that service did not know about many of the smaller roads that connect parishes. Instead, we used the road network data available for Uganda on OpenStreetMap [OpenStreetMap contributors, 2017] to model the road geography and generated approximate travel times between Parishes, using some assumptions about mean travel speeds on different types of roads. Given an estimated cost per kg per hour of transport, we could then roughly estimate the transportation cost for any given trade proposal. We estimated a travel time of less than one hour for more than 80%

of our successful trades.

There are other reasons that some of our users refused trade proposals based on their locations beyond concern that transport costs would swamp potential gains from trade. Based on user surveys, the most common additional reasons (in descending order) were: bad roads and weather; risk of being robbed; uncertainty about the trustworthiness of business partners in the new area; not having any contacts and connections in the area; the reputation of the quality of crops in some areas (some areas are known for having poor quality crops); language barriers; too hard to determine if the journey would be profitable; worries about tax rates and local competition; war, insurgency, and epidemics. (Conversely, other responses contained sentiments like the very entrepreneurial "If a trade is profitable, nothing can stop me.") Needless to say, Kudu did not model all of these concerns when judging that a match was "profitable". Even if we had elaborated our profitability model to capture more of these concerns, a further complication is that agents' preferences about all of these issues were heterogeneous and difficult to elicit.

Trust and Reputation Our system contained no mechanism for ensuring that traders honored agreements they made. Buyers could renege on previously accepted deals or renegotiate at the time of transaction by threatening to leave (which was particularly problematic in the case of perishable crops). Sellers could also attempt to renegotiate at the last minute, leveraging the fact that a buyer could not easily walk away after paying for a truck rental and driving a long distance. Escrow was a natural solution to both problems: a buyer could deposit some fraction of the trade's price into an account managed by Kudu; the system could notify the seller that the money was in place; and the buyer could tell Kudu to release the money when the goods were transferred. One issue was that this would still not have eliminated risk on the buyer side if the goods were not as promised, since the farmer would not have had to put anything in escrow; Kudu would need to mediate disputes. Various other practical hurdles made this idea challenging to implement in practice: traders could be exposed to mobile money phishing scams; kiosks to withdraw mobile money were not yet prevalent enough in rural areas; and (probably most importantly) fees for mobile money transactions were far too high.

Another way of increasing users' trust in the system might have been to inte-

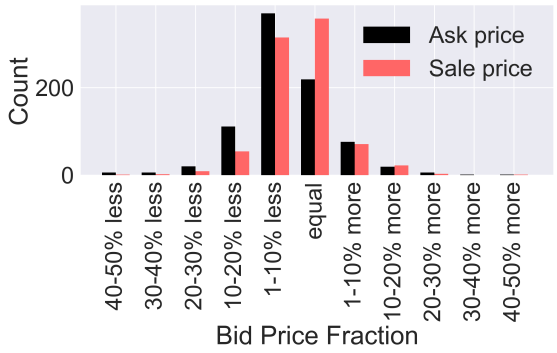


Figure 2.2: A histogram of bid prices as a fraction of ask prices and final sale prices for verified transactions. If users were truthful, no trades would ever have occurred with a bid price lower than the ask price; clearly this was not the case. However, most of the mass is distributed around the center, suggesting that user prices still conveyed useful signaling information in most cases. The buyer usually payed slightly less than their bid price in the final sale.

grate a more robust reputation system. Kudu let users blacklist anyone with whom they had a poor experience, but we could have gone further. Since we had a transaction history for every user on Kudu, we could have rewarded completion of successful matches by increasing users' priorities, and conversely deprioritized users who did not follow through. Deal coordinators could also have used such reputation information to convince users to trade with each other. Challenges to a more sophisticated reputation system were that it was expensive to verify successful trades and that "whitewashing" was easy: Kudu identities were tied to phone numbers, and new SIM cards were inexpensive. Also, while traders transact frequently, farmer usage could be more sporadic (e.g., they often made only one big sale on Kudu per season), so it would have taken a long time to develop a reliable signal of farmer reliability.

2.3.3 Matching

A significant technological challenge in the Kudu ecosystem was choosing which bids should match with which asks. We wanted to do so in a way that maximized the value that Kudu brought to the marketplace.

To quantify the value of a trade, we defined a scoring function that mapped each possible trade to a real number based on the bid and ask quantities and prices, as well as other factors such as the distance between both parties and reputation considerations. Our aim was for the scoring function to capture gains from trade. As a first attempt we set it equal to the price differential times the quantity minus a linear function of the distance between the participants, representing the cost of travel due to fuel consumption. We noticed during deployment that, because users bid and bargained strategically, this scoring function sometimes assigned negative values to trades that actually went through in practice (and therefore must actually have generated positive gains from trade). This strategic behavior is evident in Figure 2.2, which shows that users sometimes traded despite notionally negative gains. We therefore refined our implementation to assume that bid and ask prices were merely signals rather than binding constraints. We used these declared prices to fit probability distributions that modeled each user's true price, and then used these to calculate potential gains from trade. That is, we sampled from both price distributions, rejected samples where the bid price was lower than the ask price, and computed the expected gains.

We employed two separate strategies for choosing matches. Our initial solution involved deal coordinators accessing our database to make *manual matches*. We then migrated to a *hybrid* of manual matches and *automatic matches*, proposed by an optimization algorithm.

Manual Matching

When the number of bids and asks was low, an effective method for clearing the market was via *manual matching*: having deal coordinators manually look at the database of bids and asks and decide which parties to match. Kudu employed five deal coordinators, each specializing in particular treatment districts and local languages.

Such a system leveraged human intelligence and human relationships in the matching process. Deal coordinators developed intuition about which parties made good matches. This intuition was based in part on features that were hard to quantify, e.g., the personalities of the trading participants. Furthermore, deal coordinators interacted repeatedly with the same parties, developing valuable social capital and trust. A participant who had a personal relationship with a deal coordinator might have been more willing to submit bids or asks to Kudu than one who only interacted with the system electronically. (An illustration: one coordinator told us that her conversations typically involved an initial discussion about the participant's family before any discussion of the potential match.)

However, the downsides to manual matching were significant. First, it required (expensive) human employees, the number of which needed to scale at least linearly with the number of participants on the platform. (Observe that the number of potential matches grows superlinearly, even after geographic and other constraints are taken into account.) Second, it was unlikely that deal coordinators were effective at optimizing a global objective such as overall gains from trade. Discussions with the deal coordinators suggested they followed a local greedy heuristic, selecting a single buyer and then searching for the best seller that might match with that buyer. The selection of the buyer was based on their estimate of how likely the buyer was to accept a trade at that given time. The selection of the seller was then entirely based on the value of the seller to that buyer. Importantly, this process ignored the value of the seller to other potential buyers. Finally, as the system grew, search frictions, like the size of the database and the limited sorting tools, made it difficult for deal coordinators to find the best matches, even according to their own metrics.

Automatic and Hybrid Matching

An *automatic matching* algorithm takes as input a set of bids and asks and algorithmically proposes trades. Deal coordinators follow up on the trades recommended by the algorithm. In a *hybrid matching* system, deal coordinators revert to manual matching once all automatic matches have been processed. Our hope was that, as our automatic matching component improved, the participants would have been be

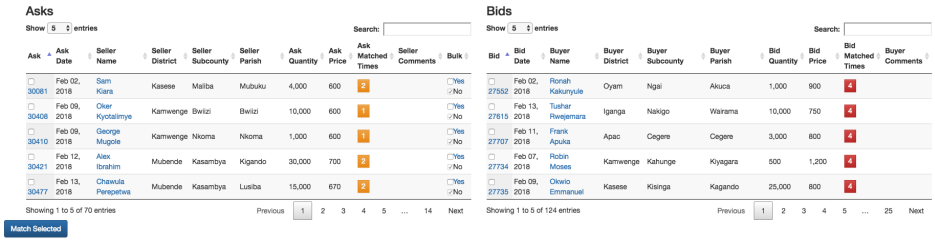


Figure 2.3: The manual matching interface on Kudu. Asks are shown on the left and bids on the right. A deal coordinator selected one from each column to create a match. This approach began to break down when the number of bids and asks grew large.

able to carry out proposed matches without intervention by the deal coordinators. Such a system would have scaled easily as the market grew, could have optimized global objectives, and would not have been significantly hindered by search frictions. However, a fully automatic system would have sacrificed the human intelligence and social capital of the deal coordinators.

Our initial 2013 pilot ran a heuristic algorithm that periodically went over all of the bids in the system in an arbitrary order and matched each bid with an unmatched ask having a high score according to our scoring function (see Ssekibuule et al. [2013]). This approach addressed the issue of facilitating search for deal coordinators. However, because it did not always intelligently choose the order in which bids were processed, it did not optimize gains from trade. Furthermore, it was not able to leverage deal coordinators' background knowledge, forcing them to concentrate on automatically selected matches.

In 2015, we introduced an improved match optimization algorithm, which ran three times a day. At run time, the algorithm simultaneously considered all bids and asks in the system and proposed a feasible set of trades that maximized the total gains from trade, according to our scoring function. (This amounted to running a maximum weight matching algorithm in a bipartite graph; the optimization could thus be performed efficiently.) Our solution also attempted to help the participants find a "fair" price. We set the recommended price of a transaction to the minimum competitive (i.e., Walrasian) prices for the matching market [Gul and Stacchetti, 1999], making truthful bidding a dominant strategy for buyers but giving farmers

incentives to manipulate their sale prices.[6] In our idealized market, buyers and farmers would have traded at our recommended price. In reality, buyers and farmers typically negotiated prices outside the system, and the recommended price was not even communicated to participants when the deal coordinators found it unhelpful.

Our automated matching system was nowhere near as successful as the manual matching system in terms of producing deals. By comparing the workflow of automatic matching to manual matching, we identified a large problem[7] that our system faced: as we proposed trades only three times per day, many participants faced long wait times before matching. Additionally, many of these matches quickly unraveled: most commonly when a deal coordinator called the seller and found out that their ask was no longer valid (see Section 2.3.2). There was little the system could do for the matched buyer in such cases—even if they had a strong bid, the algorithm would likely already have matched the strongest other asks to other bids.

In November 2017, we shifted to an automatic system, called *Kudu AI*, that offered matches continuously. The system assigned a priority to each buyer, equal to the highest potential surplus from a trade involving that buyer. When a deal coordinator entered the system, she was presented with a list of buyers, sorted by priority. Our intention was that the deal coordinator would choose to work with the highest-priority buyer in this list[8]. Once the deal coordinator selected a buyer, the algorithm selected five possible sellers for the buyer, in decreasing order of gains from trade. Deal coordinators could accept or reject any seller. Rejections came with reasons that help us improve the algorithm. When used as intended, this process mimics the greedy algorithm for maximum matching and hence captures a constant fraction of the gains from trade in a static system. Furthermore, it gave

[6] As demonstrated by a celebrated theorem due to Myerson and Satterthwaite [Myerson and Satterthwaite, 1983], it is impossible to make truthful bidding a dominant strategy for both sides of the market. We focused on incentivizing buyers because they typically constituted the short side of our market and because they had access to a more robust array of outside options.

[7] Another potential problem with this and the 2013 system was the interplay of manual and automatic matches: we worried during manual matching periods, the deal coordinators might cherry-pick the best matches, leaving less for the automatic matching to work with.

[8] The deal coordinator could also search the system for a specific buyer; this would defeat the global optimization guarantees.

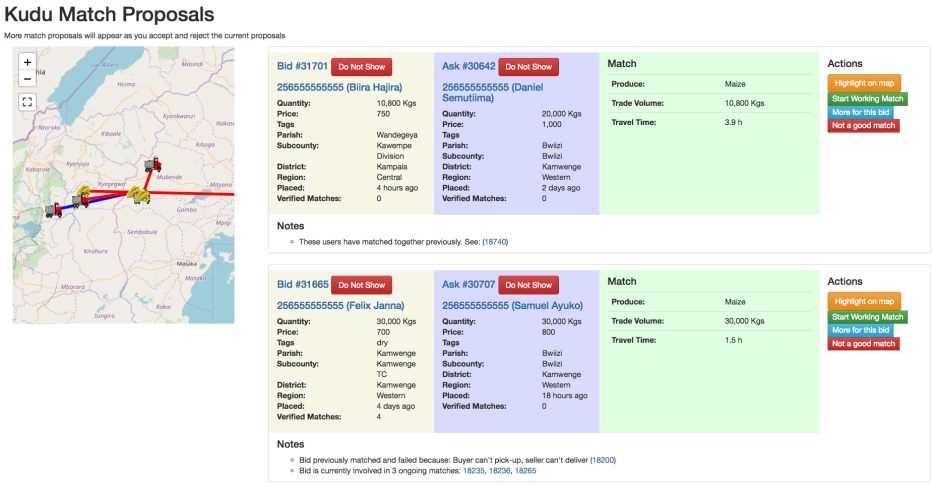


Figure 2.4: The Kudu AI interface. Deal coordinators were shown matches corresponding to the largest gains from trade. Matches were overlaid on a map of Uganda. Characteristics of the match and the involved users were also shown: for example, we highlighted if the two users have matched together before or are engaged in any ongoing transactions. Deal coordinators could accept or reject proposals, refreshing the available choices.

deal coordinators more flexibility, in the form of rejecting sellers or processing buyers in an order other than the recommended one. We hoped to use this flexibility to improve our algorithm's scoring function by leveraging the human intelligence of the deal coordinators. Ultimately, the project went dark before we were able to thoroughly learn from this interface.

2.3.4 Facilitating Trade

We partnered with AgriNet, one of Uganda's largest private-sector brokerage companies, to promote Kudu with farmers and facilitate trades with on-the-ground services. As part of this collaboration, AgriNet rolled out their agent model into the communities in which we introduced Kudu. Agents promoted Kudu by advertising the service to both farmers and local traders, typically via house-to-house visits and announcements via loudspeaker in markets. They then followed up this ad-

vertisement with a village-based meeting in which they provided information on Kudu services and training on how to use the system. Agents also distributed their phone numbers so that users could call them with questions about the service or if they needed help registering an ask or bid later in the season.

In addition to promotion and training, AgriNet offered several additional services designed to address issues that could hinder transactions between buyers and sellers, even once they had found each other on the Kudu platform. First, because many farmers in Uganda operate at a small scale, surpluses are often quite diffuse, and aggregation is necessary to attract large national buyers [Sitko and Jayne, 2014]; this requires both coordination and access to capital. Kudu had the capacity to note and electronically bulk lots of the similar crops available for sale in nearby locations; AgriNet agents were available to provide on-the-ground coordination of this bulking. In order to finance this bulking, AgriNet offered its agents access to Cash on Bag (COB) credit. Agents in turn could use this credit to pay cash-constrained farmers for 50% of the value of their crop upon bulking with the agents and 50% upon sale to the buyer.[9]

Another challenge that may have limited buyers' willingness to trade on the platform was the risk inherent in directly trading with farmers in remote villages with whom they had not yet developed trust. Buyers needed to make up-front investments in transportation out to rural villages without any guarantee that agreements made in advance regarding quantity or quality of available crops would be carried out as promised once they arrived. Buyers sometimes instead choose to trade only with trusted brokers or other traders with whom they have had repeated interactions, resulting in a fractured chain of many short-distance, relationship-based exchanges [Fafchamps and Minten, 1999]. To address these risks, AgriNet offered a "transaction guarantee" service. This service, designed to reduce the risk to buyers inherent in engaging in a more anonymous marketplace, offered transport cost compensation for buyers who traveled to a rural sale point and were disappointed.

As a further measure to address the risks involved with remote trading, local monitoring agents were used to certify the details of the transactions. After receiv-

[9] In addition to this bulking procedure, we had future plans to implement automated bundling as part of the Kudu matching process.

ing a call from a deal coordinator about an agreed-upon deal, a monitoring agent visited the seller to check the quality and quantity and communicate his findings to the buyer. The agent was present at physical transaction and oversaw exchange of money, providing regular updates to deal coordinators throughout the transaction process. These monitoring agents provided other services as well, such as recruiting and training new users through visiting local markets and village promotion meetings. Finally, the monitoring agents helped smooth out price negotiation by being physically present. For example, we have heard from deal coordinators that there was sometimes a tension after both parties had exchanged phone numbers regarding who would call the other first, out of fear of looking desperate. The monitoring agent could help address this issue by mediating the negotiation.

2.3.5 Price Information

Soliciting bids and asks on Kudu was challenging due to technological and informational constraints. We have already discussed technological constraints, which were largely beyond our power to change: e.g., few farmers have smart phones, which forced us to solicit information through limited interfaces; many are illiterate, which forced us to rely on a call center. The key informational constraint was that users were reluctant to participate in the system without knowing current market rates for the crops they were interested in trading. To tackle this challenge, we provided price quotes. Ideally our price information would have come from verified transactions that had occurred on Kudu, but our system was too small to consistently have sufficient data in enough districts to be useful.

A next hope would have been to use the bids and asks, which were more plentiful than verified transactions. However, we expected the bids and asks we received to be somewhat biased, since our platform existed in the context of outside options. Our users were only interested in using the platform if it could get them a better deal than they could otherwise find. This meant that ask prices were usually inflated and bid prices were usually shaved. Broadcasting this data could have been particularly problematic because it might have created a harmful feedback loop in which farmers received overly optimistic price information from Kudu and then priced their crops accordingly.

As a result, an initial version of our price information system simply reported the median ask price, over the previous week, for a given crop (by default, nationally, but optionally scoped to a given location). Our final implementation of this service used biweekly survey data to determine market prices for select crops in the treatment districts and reported the 25–75 percentile of wholesale prices at markets. Collecting data in this way yielded coarse information, was expensive, and resulted in stale quotes. It was nevertheless highly popular among our users.

2.4 Resource Requirements

Kudu's software requirements were minimal—it ran on a virtual machine with 2 CPUs and 4GB of memory. It was built as a Django [Django Software Foundation] web application and did not utilize any commercial software. Matching algorithms came from the iGraph [Csardi and Nepusz, 2006] library. As described in Section 2.3.2, OpenStreetMap [OpenStreetMap contributors, 2017] was used for distance calculations in our transportation cost model. Costs to run the market (prior to any wraparound services) consisted of a salary for the platform manager, short-code fees, and some radio ads.

The operating costs for running the platform from May 2015 to March 2018 (during the randomized control trial) including all of the wraparound services were $927,190, with most of this going towards salaries for deal coordinators, call center operations, monitoring agents, and other program staff. More detailed accounting is available in Bergquist et al. [2021].

2.5 Field Evaluation

Over the course of our operations, users submitted nearly 30,000 asks and over 30,000 bids, resulting in more than 850 verified completed transactions involving over 5,000 tons of grain with a value of more than $1.9 million USD.

Figure 2.6 shows the cumulative value of transactions on our platform, broken down by crop. Figure 2.7 shows all verified transactions on Kudu, plotted geographically. Figure 2.5 illustrates the users active on our platform over time, separating existing and new users.

Kudu was embedded within an extraordinarily large scale randomized control

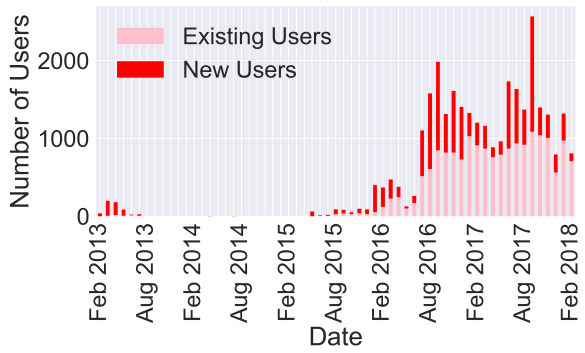


Figure 2.5: Active users over time. Each bar represents the number of unique active users in a one month interval. New users that month are highlighted. An active user is defined as a user who used any of Kudu's services during the given month. In total, 11,861 unique users engaged with the platform.

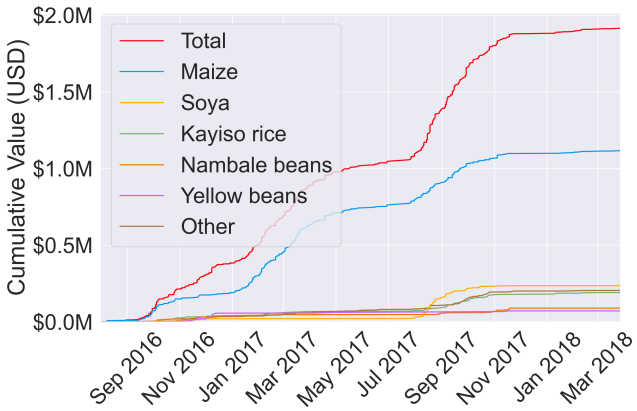


Figure 2.6: Cumulative value of verified transactions between September 2016 and March 2018.

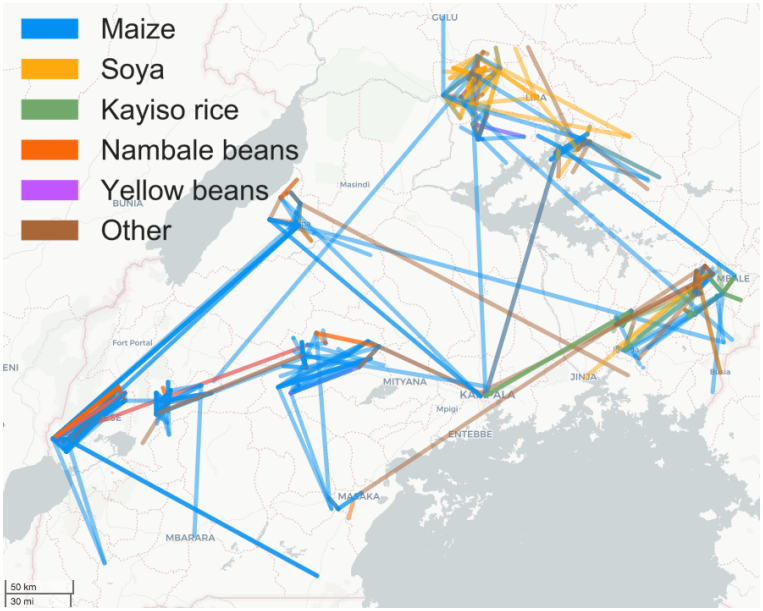


Figure 2.7: Geographical range of our verified transactions, with edges linking the reported parishes (small villages) of buyers and sellers respectively. About a third of verified transactions occurred within the same parish and are not visible on this map. The large clusters correspond to the 11 districts throughout Uganda in which Kudu was supported through in-village services. We note that Kudu spread beyond these treatment districts.

trial conducted between May 2015–March 2018, spanning six agricultural seasons [Bergquist et al., 2021]. The trial encompassed 11 districts, amounting to 110 subcounties and 236 trading centers. Half of the subcounties were reserved as a control, the other half were given the in village services described in the sections above. The study also recruited informants that gathered market prices in 236 markets every two weeks for four major crops (maize, nambale beans, matooke bananas, tomatoes). These prices were blasted to a subset of the treated farmers and to all treated traders. Impacts on farmers and traders were gauged through extensive surveys before, during, and after the intervention. Surveys tracked 1,457 traders[10] and 2,971 farming households). Farmers were stratified by their proxim-

[10]This is a high degree of coverage: 83% of the traders meeting the survey criteria in the study's

ity to a marketplace.

We refer the reader to Bergquist et al. [2021] for detailed results of the study, but share some insights here:

- By far, most activity on the platform from study participants came from traders, both on the ask and bid sides. Our platform was mainly used to arrange sales between traders, not to arrange farmgate sales with smallholder farmers as we had initially envisioned. The quantities that smallholder farmers had available to sell were typically too small to be of interest to their counterparties on the other side of the market.

- Trade flow between treated markets increased and nearby markets saw a reduction in price dispersion of 8% and 15% as one and both markets were treated respectively.

- Simply blasting SMS price information without providing the rest of the services was ineffective. Facilitating trade was essential for impact.

Perhaps the most interesting output of the study is a back-of-the-envelope welfare calculation for Kudu. It works as follows: Based on survey data, the intervention reduced trader profits, amounting to a net harm of $1.53M. When the costs of running the intervention are added, the social cost of the platform was estimated to be $2.92M. Revenue effects were then estimated for farmers by regression using the following features: whether or not they lived in a treated subcounty, lived near a marketplace, whether they were receiving SMS blasts[11], and interaction terms. The computation found a $124K welfare benefit to farmers in the study sample, at first glance much less than the costs. However, if those benefits are extended to non-surveyed households of which they should be representative in treated subcounties (which can be justified given evidence that the intervention moved trade volumes and prices), then the calculation is dominated by the 919,697 households that are "far" from a marketplace and did not receive SMS blasts. Such households were estimated to each receive a $12.29 benefit for each of the three years of the study period, leading to total benefits of $34M and a net benefit of over $30M.

districts.

[11] Only possible when living in a treated subcounty.

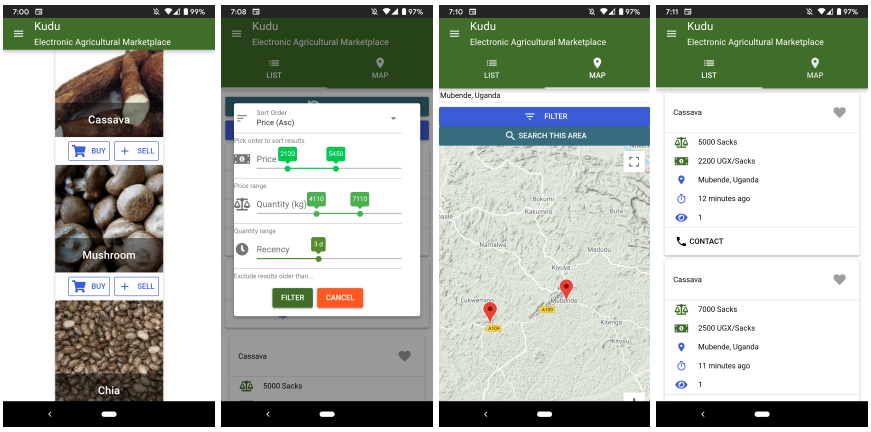


Figure 2.8: Screenshots of the Kudu application running on an Android smartphone. Users select from a menu of crops and can filter on features of the listing such as price, quantity, recency, and geography. Users interested in a listing receive the owner's contact deals and manually initiate a trade.

While we caution that this is a back-of-the-envelope analysis using non-statistically significant values (the standard error on the $12.29 benefit is $26.66), the important observation is simply that there are so many farmers, even improving their situation a small amount ($0.79/year) would have positive overall welfare effects.

2.6 Redesigning the Market To Minimize Human Intervention

Funding for Kudu ended in March 2018. Without a call center or deal coordinators, market activity ceased. Sustaining Kudu required commercial viability, which proved to be a tricky puzzle: we had no practical way to implement commissions (we could not verify transactions without a physical agent present at point-of-sale, and most deals were made in cash so there was no money flowing through the platform from which we could have extracted a cut). Furthermore, we did not expect users to tolerate listing fees: Bergquist et al. [2021] estimate the required break-even fee even for a bare-bones Kudu with no wraparound services at a dollar per listing, a value small relative to the scale of most transactions but certainly too high for farmers. A final business model we considered would be be to sell up-to-date

price information derived from Kudu's database, but this would only have been possible if Kudu was already operating at a scale large enough to reliably generate data.

We reflected on how we might revive our marketplace with significantly lower operating costs, having concluded that commercialization was unlikely. Our biggest expenses were employee salaries for deal coordinators, the call center, and monitoring agents. These services gave our brand credibility and made interaction with our system possible for many users for whom it would otherwise have been too technologically complex. Since a fully automated system was impractical for our target users, our goal was not to eliminate it entirely but rather to minimize its size. Betting on smartphones slowly penetrating the market, and in acknowledgement of the heterogeneity and complexity of user preferences, we designed an application that allows users to browse and create listings themselves. As discussed in Section 2.2, while conceptually these services could be offered through a complex USSD application on a feature phone, practically we believe that a smartphone UI is required for users to effectively search through listings given their lack of substitutability due to factors like location. The application allows users to filter products by quantity, price, and recency. Instead of placing "bids", users specify search criteria and receive notifications when new matching asks are posted to the platform. We envisioned this application being used by traders or large aggregators; a tiny call center could support smallholder farmers who would place their listings over SMS. Screenshots of the application are provided in Figure 2.8.

We released the application on the Play Store and began a very small (hundreds of dollars) marketing campaigns via Facebook advertisements to get a small number of users on the platform. Our plan was to work through initial bugs and then expand the scope of the advertising campaign as well as use radio ads to recruit more users. Our initial Facebook ad campaign had limited impact: we were not able to create enough thickness for the market to be useful; we never moved on to radio ads. Without a local partner it was difficult to gain much insight into how the app was being used and how we might promote it. We met with several candidate organizations but never arrived at a partnership. Around March 2019, as the chaos of COVID-19 hit, we stopped actively working on the application. We have posted our source code publicly at https://github.com/newmanne/KuduApp; we hope that

interested end users and/or researchers will build upon it in the future.

2.7 Lessons Learned

When we began this journey, we believed that it was most critical to settle on the right market design and to back the market with an effective clearing algorithm. We planned for market growth that would render it impossible to manually intervene in each transaction and often discussed the exciting AI challenges we would soon face: How machine learning on large datasets of successful and unsuccessful proposals would allow us to replace our hand-crafted model of gains from trade; how we would embed travelling salesman problems into our matching framework to bundle several small nearby asks; or how we should best implement reputation systems. We have since come to appreciate the importance of additional issues that have little to do with market design: identifying reliable mobile operators; balancing our local partner's competing interests against our own; tensions between wanting to act like an agile startup and ensuring the sanctity of a randomized control trial; and the difficulties of working on a problem that was, for most authors, located far afield.

The last point cannot be overstated: we often had little visibility into how the software was being used on the ground and in retrospect could probably have learned a lot more by more frequently checking in with our deal coordinators, who were much more knowledgeable regarding cultural norms (for example, an expectation that price negotiation continues right up to the point of trade) and local conditions. Our experiences with matching taught us about the effectiveness of human-in-the-loop designs, both in terms of deal coordinators' abilities to identify profitable trades but also in terms of their abilities to elicit preferences from traders. We learned that human capital is essential to running a marketplace in a location such as Uganda given the current state of technological integration. We came to appreciate the extent to which electronic markets in the developed world depend upon the ubiquity of smartphones and credit cards as we struggled with their absence.

In retrospect, we were too slow to respond to the fact that traders were our primary market participants, instead clinging to the vision that smallholder farmers

could successfully sell through our system. In general, their listings were simply too small for most traders to care about, and perhaps we should have spent fewer resources trying to popularize Kudu among this demographic. Furthermore, given that our main players were traders, we might have thought differently about the resources available to and level of sophistication of our users. One possibility is that such a trader-oriented system could still benefit smallholder farmers as they would be selling into more integrated markets, even if they did not actually use the system themselves.

Another takeaway from the project is that information is most useful when it is actionable: the randomized control trial showed that our centralized marketplace enabled significant welfare gains beyond price blasts alone. Information should also be timely: many of our earlier proposals failed because we did not appreciate that listings were highly time sensitive and required quick responses.

2.8 Conclusions

This chapter has described Kudu, an electronic market for agricultural trade in Uganda designed to combat inefficiencies in rural agricultural markets. Traders and farmers posted bids and asks using a feature phone. Kudu then proposed matches, leveraging a combination of optimization algorithms, data-driven models, and human expertise. Our system was augmented by a rich variety of support services that help to facilitate trade. The system was active for several years, involving tens of thousands of users and yielding verified trades totaling more than $1.9 million USD. Results from a multi-year randomized control trial demonstrated that Kudu and all of the wraparound services accompanying it was successful at reducing arbitrage opportunities between nearby markets. A rough welfare calculation suggests that Kudu achieved net welfare benefits.

Chapter 3

Deep Optimization for Spectrum Repacking

This chapter describes a second piece of computational infrastructure underpinning a complex, electronic market: the incentive auction. A crucial element of the auction design was a solver, dubbed SATFC, that determined whether sets of stations could be "repacked"; it needed to run every time a station was given a price quote. We first introduce the incentive auction and one of its core computational challenges: the station repacking problem. Next, we describe the process by which we built SATFC.

3.1 Introduction

We begin by describing the context in which the incentive auction arose. Many devices, including broadcast television receivers and cell phones, rely on the transmission of electromagnetic signals. These signals can interfere with each other, so transmission is regulated: e.g., in the US, by the Federal Communications Commission (FCC). Because electromagnetic spectrum suitable for wireless transmission is a scarce resource and because it is difficult for a central authority to assess the relative merits of competing claims on it, since 1994 the FCC has used *spectrum auctions* to allocate broadcast rights (see, e.g., Milgrom [2004]). Many regulators around the world have followed suit. At this point, in the US (as in many other

countries), most useful radio spectrum has been allocated.

Interest has thus grown in the *reallocation* of radio spectrum from less to more valuable uses. Spectrum currently allocated to broadcast television has received particular attention, for two reasons. First, over-the-air television has been losing popularity with the rise of cable, satellite, and streaming services. Second, the upper UHF frequencies used by TV broadcasters are particularly well suited to wireless data transmission on mobile devices—for which demand is growing rapidly—as they can penetrate walls and travel long distances [Knutson and Gryta, 2014].

It thus made sense for at least some broadcasters to sell their licenses to wireless internet providers willing to pay for them. Ideally, these trades would have occurred bilaterally and without government involvement, as occurs in many other markets. However, two key obstacles made such trade unlikely to produce useful, large-scale spectrum reallocation, both stemming from the fact that wireless internet services require large, contiguous blocks of spectrum to work efficiently. First, a buyer's decision about which block of spectrum to buy would limit the buyer to trading only with broadcasters holding licenses to parts of that block; it could be hard or impossible to find such a block in which all broadcasters were willing to trade. Second, each of these broadcasters would have "holdout power", meaning the broadcaster could demand an exorbitant payment in exchange for allowing the deal to proceed. The likely result would have been very little trade, even if broadcasters valued the spectrum much less than potential buyers.

A 2012 Act of Congress implemented a clever solution to this problem. It guaranteed each broadcaster interference-free coverage in its broadcast area on *some* channel, but not necessarily on its currently used channel. This meant that if a broadcaster was unwilling to sell its license it could instead be moved to another channel, solving the holdout problem. To free up the channel that would permit this move to take place, broadcast rights could be bought from another station in the appropriate geographical area, even if this second station did not hold a license for spectrum due for reallocation. In what follows, we call such an interference-free reassignment of channels to stations a *feasible repacking*.

These trades and channel reassignments were coordinated via a novel spectrum auction run by the FCC between March 2016 and April 2017, dubbed the *incentive*

auction. The result of the auction was to remove 14 UHF-TV channels from broadcast use, to sell 70 MHz of wireless internet licenses for $19.8 billion, and to make 14 MHz of spectrum available for unlicensed uses. With fewer remaining channels for TV stations, the TV spectrum needed to be reorganized; not all stations could be reassigned channels in the compressed TV band. Each station was either "repacked" in the leftover channels or voluntarily sold its broadcast rights, either going off the air or switching to a lower-quality band. These volunteers received a total of $10.05 billion to make repacking possible by yielding or exchanging their rights.

Roughly, the reverse auction began with a *clearing target* or number of TV channels to decommission. Stations were approached one at a time with a series of decreasing price offers for their broadcast rights. When a station refused an offer, it exited the auction irrevocably and was guaranteed a spot in the leftover channels. As prices fell and more stations declined offers, the leftover channels became more and more congested. Before any station's bid was processed, a "feasibility checker" ensured that the station could still fit in the leftover channels alongside the exited stations without causing undue interference; if it could not, that station was "frozen", meaning that its price stopped falling and it was no longer eligible to exit the auction. The reverse auction further included a provision that allowed stations to exchange their broadcast rights for both a channel in a less desirable (VHF) band and monetary compensation. After the reverse auction concluded, a "forward" ascending clock auction was used to sell licenses in the cleared spectrum to mobile carriers. An outer-loop procedure iterated between reverse and forward auctions to identify the largest possible clearing target for which forward auction revenues exceeded costs.

The problem of checking the feasibility of repackings is central to the reverse auction, likely to arise tens of thousands of times in a single auction. Unfortunately, this problem is NP-complete, generalizing graph coloring. The silver lining is that interference constraints were known in advance—they were derived based on the locations and broadcast powers of existing television antennas—and so it was reasonable to hope for a heuristic algorithm that achieved good performance on the sorts of problems that would arise in a real auction. However, identifying an algorithm that would be fast and reliable enough to use in practice remained

challenging. Since each feasibility check depends on the results of those that came before—if a station is found to be frozen, it cannot exit—these problems must be solved sequentially. Time constraints for the auction as a whole required that the auction iterate through the stations at least twice a day, which worked out to a time cutoff on the order of minutes. It was thus inevitable that some problems would remain unsolved. Luckily, the auction design is robust to such failures, treating them as proofs of infeasibility at the expense of raising the cost required to clear spectrum.

This chapter describes our experience building SATFC 2.3.1, the feasibility checker used in the reverse auction. We leveraged automatic algorithm configuration approaches to derive a portfolio of complementary algorithms that differ in their underlying (local and complete) search strategies, SAT encodings, constraint graph decompositions, domain-specific heuristics, and use of a novel caching scheme.

We use the term "deep optimization" to refer to this approach,[1] with the goal of emphasizing its conceptual similarity to deep learning. Classical machine learning relied on features crafted based on expert insight, model families selected manually, and model hyperparameters tuned essentially by hand. Deep learning has shown that it is often possible to achieve substantially better performance by relying less on expert knowledge and more on enormous amounts of computation and huge training sets. Specifically, deep learning considers parametric models of very high dimension, using expert knowledge only to identify appropriate invariances and model biases, such as convolutional structure. (In some cases it is critical that these models be "deep" in the sense of having long chains of dependencies between parameters, but in other cases great flexibility can be achieved even with models only a couple of levels deep; e.g., Zhang et al. [2016].) We argue that a

[1]There exists a large body of prior work that investigates the use of algorithm configuration to design novel algorithms from large, parameterized spaces; we believe, however, that the work described in this chapter is the most consequential application of such techniques to date. Much of the literature just mentioned focuses on algorithm configuration tools [Ansótegui et al., 2009, Hutter et al., 2009, 2011, Kadioglu et al., 2010, López-Ibáñez et al., 2016, Xu et al., 2010] (which we take as given in this chapter) rather than algorithm design methodology. Most work in the latter vein either addresses the much broader problem of algorithm synthesis (e.g., Di Gaspero and Schaerf [2007], Monette et al. [2009], Westfold and Smith [2001]) or defines the overall approach only implicitly (e.g., KhudaBukhsh et al. [2016]). The most prominent exception is "programming by optimization" [Hoos, 2012]; however, it emphasizes connections to software engineering and does not limit itself to parametric design spaces.

similar dynamic applies in the case of heuristic algorithms for discrete optimization, which aim to achieve good performance on some given dataset rather than in the worst case. Traditionally, experts have designed such heuristic algorithms by hand, iteratively conducting small experiments to refine their designs. We advocate an approach in which a computationally intensive procedure is used to search a high-dimensional space of parameterized algorithm designs to optimize performance over a large set of training data. We aim to minimize the role played by expert knowledge, restricting it to the identification of parameters that could potentially lead to fruitful algorithm designs. We also encourage deep dependencies via chains of parameters each of whose meaning depends on the value taken by one or more parents.

Overall, this chapter demonstrates the value of the deep optimization approach via the enormous performance gains it yielded on the challenging and socially important problem of spectrum repacking. After formally stating the station repacking problem and the incentive auction rules, we define our large algorithm design space and the search techniques we used. We assess the results on problems that arose in runs of our reverse auction simulator, investigating our solver's runtime. In Chapter 4 we assess the solver's impact on economic outcomes.

3.2 The Incentive Auction

In this section we describe the reverse auction and show where it fits within the context of the incentive auction. We begin by introducing the station repacking problem. Solving repacking problems is a key subroutine within the reverse auction loop. We then proceed to the auction rules.

3.2.1 Station Repacking

We now describe the station repacking problem in more detail. [2]

[2]Similar problems have been studied in other contexts, falling under the umbrella of *frequency assignment problems*. See e.g., Aardal et al. (2007) for a survey and a discussion of applications to mobile telephony, radio and TV broadcasting, satellite communication, wireless LANs, and military operations. We are not aware of other published work that aims to optimize feasibility checking in the incentive auction setting.

Prior to the auction, each television station $s \in S$ in the US and Canada[3] was assigned to a channel $c_s \in \mathcal{C} \subseteq \mathbb{N}$. The set of channels $\mathcal{C}$ can be partitioned into three equivalence classes, referred to as *bands*. Listed in decreasing order of desirability, these bands are: UHF (channels 14–51), high VHF ("HVHF", channels 7–13) and low VHF ("LVHF", channels 1–6). We use pre(s) to refer to a station's pre-auction band, sometimes called a station's home band.

Each station was only eligible to be assigned a channel on a subset of $\mathcal{C}$, given by a *domain* function $\mathcal{D} : \mathcal{S} \to 2^{\mathcal{C}}$ that maps from stations to these sets. The FCC determined pairs of channel assignments that would cause harmful interference based on a complex, grid-based physical simulation ("OET-69" [FCC, 2013]); this pairwise constraint data is publicly available [FCC, 2015b]. Let $\mathcal{I} \subseteq (\mathcal{S} \times \mathcal{C})^2$ denote a set of *forbidden station–channel pairs* $\{(s, c), (s', c')\}$, each representing the proposition that stations s and s' could not concurrently be assigned to channels c and c', respectively.

The goal of the incentive auction was to remove some broadcasters from the airwaves and assign the remaining stations new channels from a reduced set $\overline{\mathcal{C}} = \{c \in \mathcal{C} \mid c < \overline{c}\}$. This reduced set is defined by $\overline{c} \in \mathcal{C}$; each choice of $\overline{c}$ corresponds to some clearing target. The actual incentive auction ended with $\overline{c} = 37$, allowing the higher numbered channels to be used for other purposes.

A *feasible assignment* is a mapping $\gamma : \mathcal{S} \to \overline{\mathcal{C}}$ that assigns each station a channel from its domain that satisfies the interference constraints: i.e., for which $\gamma(s) \in \mathcal{D}(s)$ for all $s \in S$, and $\gamma(s) = c$ implies that $\gamma(s') \neq c'$ for all $\{(s, c), (s', c')\} \in \mathcal{I}$. As it turns out, interference constraints come in two kinds. *Co-channel constraints* specify that two stations may not be assigned to the same channel; *adjacent-channel constraints* specify that two stations may not be assigned to some pair of nearby channels. Thus, forbidden station–channel pairs are always of the form $\{(s, c), (s', c + i)\}$ for some stations $s, s' \in \mathcal{S}$, channel $c \in \mathcal{C}$, and $i \in \{0, 1, 2\}$. Furthermore, no interference constraint involves channels in more than one band.

Lastly, we define the *interference graph* as an undirected graph in which there is one vertex per station and an edge exists between two vertices s and s' if the corresponding stations participate together in any interference constraint: i.e., if there

[3]Canadian stations did not bid in the auction but could be reassigned new channels.

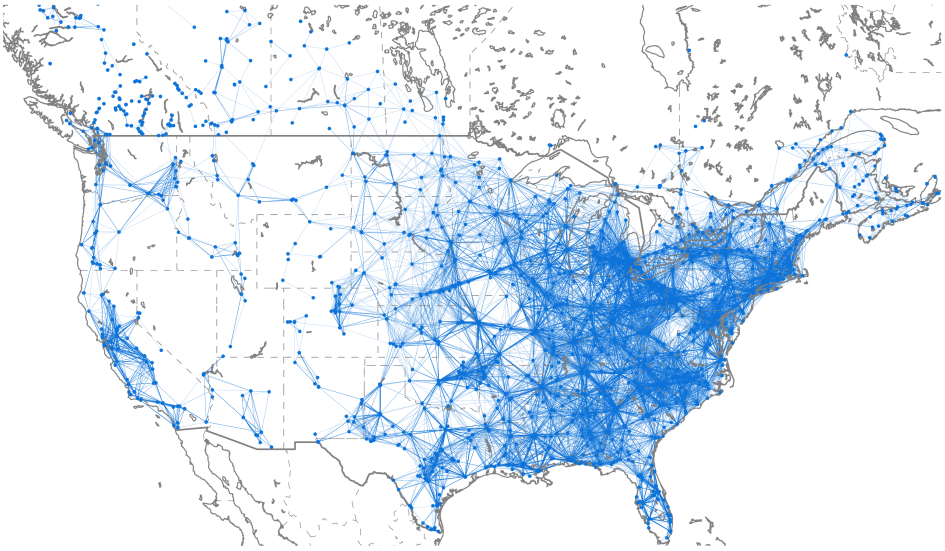


Figure 3.1: Interference graph derived from the FCC's constraint data.

exist $c, c' \in C$ such that $\{(s, c), (s', c')\} \in I$. Figure 3.1 shows the interference graph for the US and Canada.

3.2.2 Reverse Auction

We describe a simplified version of the reverse auction in which only the UHF band is repacked: only UHF stations participate in the auction and the only possible outcomes for each station are going off-air or continuing to broadcast in UHF. The real auction also repacked two VHF bands, but the inclusion of these bands complicates things significantly; we briefly sketch some details in Section 3.2.3 and further elaborate on pricing VHF options in Appendix A.2.1. The complete set of auction rules was published by the FCC in a 230-page document [FCC, 2015a].

We begin by describing the reverse auction at a high level before giving more details about various key elements. First, stations respond to opening prices and decide whether to participate in the auction. Next, a solver finds an initial feasible channel assignment for all non-participating stations to minimize the number of channels required for those broadcasters, setting an initial clearing target $\overline{c}$. The auction then attempts to buy broadcast rights as necessary so that all stations re-

maining on air can fit into the available channels. It proceeds over a series of rounds, which consist of: (1) decrementing the clock and determining new prices, (2) collecting bids and (3) processing bids. We provide pseudocode for the reverse auction as Algorithm 3 in Appendix A.2.

A forward auction to sell the cleared spectrum follows the reverse auction. If sufficient revenue is raised in the forward auction, the incentive auction terminates; otherwise, the reverse auction resumes with a lower clearing target. We elaborate on each step of the reverse auction below.

Prices

Prior to the auction, the FCC used a *scoring rule* to assign a *score* (also sometimes referred to as a *volume*) to each station, which we denote by score (s). The score was used to determine individualized opening prices and was a function of both the station's interference constraints and the population of viewers that a station reached before the auction, which we denote by Population (s). We will have more to say about scoring rules in Section 4.4.3.

The reverse auction is a descending clock auction. The initial base clock price was $p_0 = \$900$ both in the incentive auction and in our simulations. At the start of each round, the base clock price is reduced to $p_t = p_{t-1} - d_t$, where $d_t = \max(\frac{p_{t-1}}{20}, \frac{p_0}{100})$. Scores transform base clock prices to individualized station prices; this is, prices in each round $P_{s;t}$ are computed as $P_{s;t} = \text{score}(s) \cdot p_t$. We use $P_{s;\text{Open}}$ to refer to the auction's opening prices.

A "winning" station is one that ultimately goes off-air or moves to a different band. More specifically, if the final channel assignment is γ, s is winning if $\text{post}(\gamma, s) \neq \text{pre}(s)$, where $\text{post}(\gamma, s)$ returns either the band to which s is assigned under γ or OFF if s is not assigned to a band under γ. We refer to the set of winning stations as S_{winners}. Throughout the auction, we track each station's "winning price" $\mathcal{P} : \mathcal{S} \to \mathbb{R}^+$. $\mathcal{P}(s)$ is the price that would have to be paid to s if the auction were to end immediately and s was a winning station. Initially $\mathcal{P}(s) = P_{s;\text{Open}}$.

Bidding

When only the UHF band is repacked, a bid in a given round corresponds to a binary decision. A station can accept $P_{s;t}$, indicating that it prefers to relinquish its broadcast rights and receive $P_{s;t}$. Alternatively, a station can reject $P_{s;t}$, indicating that it prefers to continue to broadcast. If a station's bid to reject an offer is ever processed (as will be explained in the following section), it is said to have "exited" the auction. Such a station is never asked to bid again. An exited station receives no compensation and will continue to broadcast in its pre-auction band after the auction (albeit on a possibly different channel). We refer to the set of exited stations by S_{exited}.

Bid Processing

In the bid processing step, stations are considered one after another, in descending order of $\frac{\mathcal{P}(s)-P_{s;t}}{\text{score}(s)}$. When a station s is considered, first the feasibility checker is invoked to determine whether it is possible to repack s along with the exited stations: i.e., given a time limit, it tries to find a feasible assignment for $\{s\} \cup S_{\text{exited}}$. If the feasibility checker cannot repack s, its bid is not examined, and s is said to be "frozen". A station that is guaranteed to be frozen for the remainder of the stage is called a *provisional winner*; in a UHF-only auction, every frozen station is provisionally winning. In this case, $\mathcal{P}(s)$ is not reduced and s will no longer be asked to bid. If the feasibility checker can repack s, then $\mathcal{P}(s)$ is reduced and its bid is examined. If s bid to accept $P_{s;t}$, $\mathcal{P}(s)$ is lowered to $P_{s;t}$ and s remains "active", meaning it will be asked to bid again next round. If s bids to reject $P_{s;t}$, s permanently exits the auction.

Transitioning Between Auction Stages

A reverse auction stage ends when all stations are either frozen or have exited. Following each reverse auction stage is a forward auction stage where mobile carriers bid on licenses in the cleared spectrum. If the forward auction generates enough revenue to cover the costs of the reverse auction (the payouts to the winning sta-

tions $\sum_{s \in S_{\text{winners}}} \mathcal{P}(s)$),[4] the incentive auction terminates and each frozen station is paid $\mathcal{P}(s)$. An unsuccessful forward auction (one that does not raise sufficient revenue) triggers another stage of the reverse auction with a smaller clearing target.

The incentive auction thus determines the amount of spectrum to clear endogenously by iterating through stages of reverse and forward auctions that clear progressively less spectrum until a stage occurs in which the forward auction covers the costs of the reverse auction. When a new reverse auction stage begins, $\bar{c}$ is increased, expanding the set $\overline{\mathcal{C}}$. Given additional channels, some frozen stations may now be repackable; such stations are said to be in "catch-up" mode. At the beginning of a new reverse auction stage, the base clock p_t resets. A station in catch-up mode "unfreezes" if it can be repacked in the first round in which it would face a weakly lower price than the price at which it froze, $\mathcal{P}(s)$. Subsequent stages otherwise proceed like the initial stage.

3.2.3 Worked Example

Figure 3.2 illustrates the reverse auction through a worked example. Consider an auction setting with six stations A, B, C, D, E, F having valuations $V_A > V_B > V_C > V_D > V_E > V_F$. Assume that stations bid straightforwardly: they accept offers above their values and reject offers below their values. Let each station be identically scored (so starting prices are the same for all stations); thus, we can drop the station subscript when discussing prices, writing P_t instead of $P_{s;t}$. Assume that the feasibility checker is perfect, always finding a feasible assignment when one exists and always determining infeasibility otherwise. For convenience, let clock decrements be so small that we can model the clock as falling continuously. Let $\mathcal{C} = \{1, 2, 3\}$ and let $\mathcal{I}$ be structured so that all stations have co-channel constraints on every channel with each neighboring station according to the interference graph in Figure 3.2. Let stations B and C additionally have adjacency constraints with each other due to their close proximity, prohibiting them from jointly broadcasting on adjacent channels.

Let $\bar{c} = 2$ initially, so that the auction begins trying to repack the stations

[4] Technically, the forward auction had to generate about $2 billion more: it also had to cover FCC expenses and the estimated costs of station retuning. In our examples and experiments, we ignore these additional requirements and only focus on the payments to winning stations.

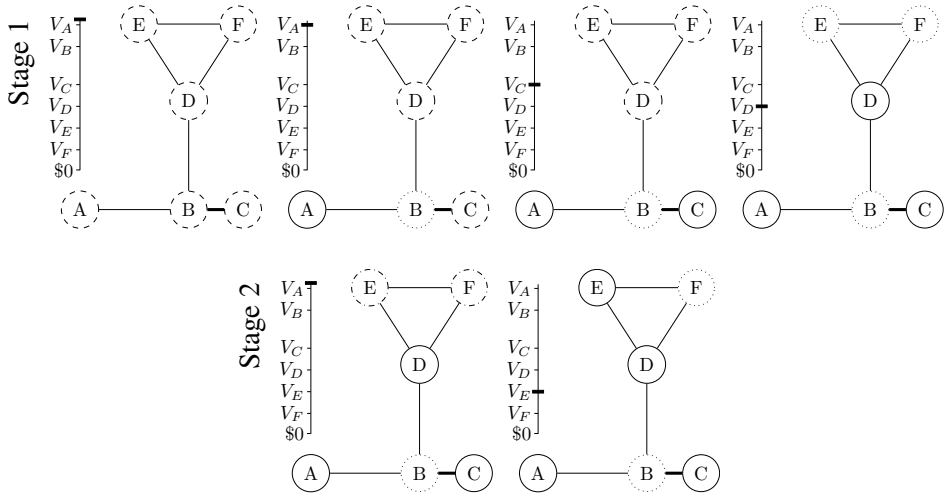


Figure 3.2: An illustration of the reverse auction example described in Section 3.2.3. Each row corresponds to a stage, and a new figure is drawn each time a station exits. Active stations are dashed, exited stations are solid, frozen stations are dotted, and stations with catch-up status are dashdotted. Connected stations cannot jointly broadcast on the same channel. B and C additionally cannot jointly broadcast on adjacent channels (shown by the thick bold edge between them). To the left of each figure is the current clock price.

into a single channel. Near the high opening prices, stations bid to remain off-air and their bids are processed because each station can exit. Nothing changes until $P_t < V_A$, at which point station A exits the auction. This movement freezes station B at price $\mathcal{P}(B) = V_A$, since A and B cannot both broadcast on the single channel. Other stations remain bidding and prices continue to fall until $P_t < V_C$ and C exits the auction. C's exit does not impact the other stations, so they continue to bid until $P_t < V_D$. At this point, D exits the auction, freezing E and F each $\mathcal{P}(E) = \mathcal{P}(F) = V_D$. Every station is now either frozen or has exited, so the stage ends. Stations B, E, and F are frozen at the end of the stage. If the incentive auction does not proceed to another stage, the total value removed from the airwaves will be $V_B + V_E + V_F$ and the total cost of freeing up two channels will be $V_A + 2V_D$.

Let us assume that in the forward auction, wireless carriers are unwilling to pay the repacking cost of $V_A + 2V_D$ to repurpose two channels worth of spectrum. The

incentive auction then proceeds to a second stage where $\overline{c} = 3$: only one channel is cleared and two channels remain for repacking stations. Stations E and F can now be repacked alongside the exited stations and so enter catch-up mode. B cannot be repacked and remains frozen. P_t resets to a high value and then descends until $P_t < V_D$. At this point, E and F both transition from catch-up mode to bidding, since V_D was the price at which they froze. The auction continues until $P_t < V_E$, at which point E exits and freezes F at a price $\mathcal{P}(F) = V_E$. Again, all stations are either frozen or exited, so the second stage completes. Assuming the ensuing forward auction raises sufficient revenue, stations B and F will removed from the airwaves for a value loss of $V_B + V_F$ and a cost of $V_A + V_E$.

Repacking VHF

We conclude by noting some of the reverse auction modifications when the VHF band is also repacked and stations can additionally bid to move between bands. Let $b_{s,t} \in B_{s,t}$ represent station s's bid in round t, where $B_{s,t} \subseteq \{\text{OFF, LVHF, HVHF, UHF}\}$ denotes the options available to s in round t. To reduce strategic options available to bidders, bidding is a restricted by a "ladder constraint": Stations can never move "down" the ladder of bands (ordered as in the list above, with UHF at the top and OFF at the bottom) and can never go higher than their home band. For example, a station with a home band of HVHF currently assigned to OFF could bid to remain in OFF, move to LVHF, or exit to HVHF, but would not be allowed to move to UHF since UHF is above its home band. If the station were instead assigned to LVHF, it could bid to remain in LVHF or exit to HVHF; it would not be able to bid for OFF because OFF is below LVHF, and it would still not be able to bid for UHF because UHF is above its home band. In each round, each bidding station is offered a separate price for each of its legal movements. We use $P_{s;b;t}$ to represent the price offered to station s in round t for selecting band b. We discuss how prices are computed in Appendix A.2.1. A final difference we will mention is the introduction of fallback bids. While any number of stations can go off-air, the VHF bands have limited capacity. As a result, the auction may not be able to accommodate certain bids for moving into VHF bands. For example, consider a UHF station s that bid for going off-air in the previous round and in the current

round bids to move to HVHF. If s is not frozen when its bid is examined, the feasibility checker will try to determine if it is possible to fit s alongside the set of exited stations whose home band is HVHF. If the feasibility checker cannot find a feasible repacking, s's fallback bid is examined. This fallback bid can be either to exit or duplicate the previous bid (i.e., in the example, it would be as if s had again bid to go off-air). We denote such bids as $\text{fallback}_{s,t}$.

3.3 A Deep Optimization Approach

In the remainder of the chapter, we discuss how we tackled the station repacking problem using "deep optimization", taking a data-driven, highly parametric, and computationally intensive approach to solver design.

Why should we hope that station repacking, an NP-complete problem, can be solved effectively in practice? First, we only need to be concerned with problems involving subsets of a fixed set of stations and a fixed set of interference constraints: those describing the television stations currently broadcasting in the United States and Canada.

Second, note that we are not interested in optimizing worst-case performance even given our fixed interference graph, but rather in achieving good performance on the sort of instances generated by actual reverse auctions. These instances depend on the order in which stations exit the auction, which depends on stations' valuations, which depend in turn (among many other factors) on the size and character of the population reached by their broadcasts. The distribution over repacking problems is hence far from uniform.

Third, descending clock auctions repeatedly generate station repacking problems by adding a single station s^+ to a set S^- of provably repackable stations. This means that every station repacking problem $(S^- \cup \{s^+\}, C)$ comes with a partial assignment $\gamma^- : S^- \to C$ that we know is feasible on restricted station set S^-; we will see in what follows that this fact is extremely useful.

Finally, many repacking problems are trivial: in our experience, problems involving only VHF channels can all be solved quickly; furthermore, the vast majority of UHF problems can be solved greedily simply by checking whether s^+ can be augmented directly with γ^-. However, solving the remaining problems is

crucial to the economic outcomes achieved by the auction. In what follows, we restrict ourselves to "non-trivial" UHF problems that cannot be solved by greedy feasibility checking.

As we show in our experiments (see Section 3.5), off-the-shelf solvers could not solve a large enough fraction of station repacking problems to be effective in practice. To do better, we needed a customized algorithm optimized to perform well on our particular distribution of station repacking problems. We built our algorithm via the deep optimization approach, meaning that we aimed to use our own insight only to identify design ideas that showed promise, relegating the work of combining these ideas and evaluating the performance of the resulting algorithm on realistic data (see Section 3.4) to an automatic search procedure.

3.3.1 The Design Space

Our first task was thus to identify a space of algorithm designs to consider. This was not just a pen-and-paper exercise, since each point in the space needed to correspond to runnable code. We focused on encoding station repacking as a propositional satisfiability (SAT) problem. The SAT formalism is well suited to station repacking, which is a pure feasibility problem with only combinatorial constraints. (It may also be possible to achieve good performance with MIP or other encodings; we did not investigate such alternatives in depth.) The SAT reduction is straightforward: given a station repacking problem (S, C) with domains D and interference constraints I, we create a Boolean variable $x_{s,c} \in \{\top, \bot\}$ for every station–channel pair $(s, c) \in S \times C$, representing the proposition that station s is assigned to channel c. We then create three kinds of clauses: (1) $\bigvee_{d \in D(s)} x_{s,d}\ \forall s \in S$ (each station is assigned at least one channel); (2) $\neg x_{s,c} \vee \neg x_{s,c'}\ \forall s \in S, \forall c, c' \neq c \in D(s)$ (each station is assigned at most one channel); (3) $\neg x_{s,c} \vee \neg x_{s',c'}\ \forall \{(s, c), (s', c')\} \in I$ (interference constraints are respected). Note that (2) is optional: if a station is assigned more than one channel, we can simply pick one channel to assign it from among these channels arbitrarily. We thus created a parameter indicating whether to include these constraints. In the end, a SAT encoding of a problem involving all stations at a clearing target of 36 involved 73,187 variables and 2,917,866 clauses.

Selecting Solvers

Perhaps the most important top-level parameter determines which SAT solver to run. (Of course, each such solver will have its own (deep) parameter space; other parameters will describe design dimensions orthogonal to the choice of solver, as we will discuss in what follows.) The SAT community has developed a very wide variety of solvers and made them publicly available (see e.g., Järvisalo et al. (2012)). In principle, we would have made it possible to choose every solver that offered even reasonable performance. However, doing so would have been too costly from the perspective of software integration and (especially) reliability testing. We thus conducted initial algorithm configuration experiments (see Section 3.3.2) on 20 state-of-the-art SAT solvers, drawn mainly from SAT solver competition entries collected in AClib [Hutter et al., 2014b]. We illustrate the performance of their default configurations in Figure 3.3; most improved at least somewhat from their default configurations as a result of algorithm configuration. We identified two solvers that ended up with the strongest post-configuration performance—one complete and one based on local search—both of which have been shown in the literature to adapt well to a wide range of SAT domains via large and flexible parameter spaces. Our first solver was `clasp` [Gebser et al., 2007], an open-source solver based on conflict-driven nogood learning (98 parameters). Our second was the open-source `SATenstein` framework [KhudaBukhsh et al., 2016], which allows arbitrary composition of design elements taken from a wide range of high-performance stochastic local search solvers (90 parameters), including `DCCA` and `gNovelty`$^+$ (of which 3 solvers in our set are variants).

Using the Previous Solution

While adapting `clasp` and `SATenstein` to station repacking data yielded substantial performance improvements, neither reached a point sufficient for deployment in the real auction. To do better, it was necessary to leverage specific properties of the incentive auction problem. Rather than committing to specific speedups, we exposed a wide variety of possibilities via further parameters. We began by considering two methods for taking advantage of the existence of a partial assignment γ^-. The first method checks whether a simple transformation of γ^- is enough to yield

a satisfiable repacking. Specifically, we construct a small SAT problem in which the stations to be repacked are s^+ and all stations $\Gamma(s^+) \subseteq S$ neighboring s^+ in the interference graph, fixing all other stations $S \setminus \Gamma(s^+)$ to their assignments in γ^-. Any solution to this reduced problem must be a feasible repacking; however, if the reduced problem is infeasible we cannot conclude anything. However (depending on the value of a parameter), we can keep searching: unfixing all stations that neighbor a station in $\Gamma(s^+)$, and so on.

Our second method uses γ^- to initialize local search solvers. Such solvers search a space of complete variable assignments, typically following gradients to minimize an objective function such as the number of violated constraints, with occasional random steps. They are thus sensitive to their starting points. Optionally, we can start at the assignment given by γ^- (randomly initializing variables pertaining to s^+). We can also optionally redo this initialization on some fraction of random restarts.

Problem Simplification

Next, we considered three preprocessing techniques that can simplify station repacking problems. First, we added the option to run the arc consistency algorithm, repeatedly pruning values from each station's domain that are incompatible with every channel on a neighboring station's domain.

Second, we enabled elimination of unconstrained stations. A station s is unconstrained if, given any feasible assignment of all of the other stations in $S \setminus s$, there always exists some way of feasibly repacking s. Unconstrained stations can be removed without changing a problem's satisfiability status. Various algorithms exist for identifying unconstrained stations; we determine this choice via a parameter. (All such stations can be found via a reduction to the polytime problem of eliminating variables in a binary CSP [Cohen et al., 2015]; various sound but incomplete heuristics run more quickly but identify progressively fewer unconstrained stations.)

Third, the interference graph induced by a problem may consist of multiple connected components; we can optionally run a linear-time procedure to separate them into distinct SAT problems. We only need to solve the component to which

s^+ belongs: γ^- supplies feasible assignments for all others. Arc consistency and unconstrained station removal can simplify the interference graph by removing edges and nodes respectively. This can shrink the size of the component containing s^+ and make this technique even more effective.

Containment Caching

Finally, we know that every repacking problem will be derived from a restriction of the interference graph to some subset of $\mathcal{S}$. We know this graph in advance of the auction; this suggests the possibility of doing offline work to precompute solutions. However, our graph has 2 990 nodes, and the number of restricted graphs is thus $2^{2990} \approx 10^{900}$. Thus, it is not possible to consider all of them offline.

Not every restricted problem is equally likely to arise in practice. To target likely problems, we could simply run a large number of simulations and cache the solution to every repacking problem encountered. Unfortunately, we found that it was extremely rare for problems to repeat across sufficiently different simulator inputs, even after running hundreds of simulations (generating millions of instances and costing years of CPU time). However, we can do better than simply looking for previous solutions to a given repacking problem. If we know that S is repackable then we know the same is true for every $S' \subseteq S$ (and indeed, we know the packing itself—the packing for S restricted to the stations in S'). Similarly, if we know that S was not packable then we know the same for every $S' \supseteq S$. This observation dramatically magnifies the usefulness of each cached entry S, because each S can be used to answer queries about an exponential number of subsets or supersets. This is especially useful because sometimes it can be harder to find a repacking for subsets of S than it can be to find a repacking for S.

We call a cache meant to be used in this way a *containment cache*, because it is queried to determine whether one set contains another (i.e., whether the query contains the cache item or vice versa). To the best of our knowledge, containment caching is a novel idea. A likely reason why this scheme is not already common is that querying a containment cache is nontrivial: one cannot simply index entries with a hash function; instead, an exponential number of keys can match a given query. We were nevertheless able to construct an algorithm that solved this prob-

lem quickly in our setting.[5] We observe that containment caching is applicable to any family of feasibility testing problems generated as subsets of a master set of constraints, not just to spectrum repacking.

In more detail, we maintain two caches, a *feasible cache* and an *infeasible cache*, and store each problem we solve in the appropriate cache. We leverage the methods from Section 3.3.1 to enhance the efficiency of our cache, storing full instances for SAT problems and the smallest simplified component for UNSAT problems. When asked whether it is possible to repack station set S, we first check whether a subset of S belongs to the infeasible cache (in which case the original problem is infeasible); if we find no matches, we decompose the problem into its smallest simplified components and check for each if the feasible cache contains a superset of those stations, in which case the original problem is feasible.

3.3.2 Searching the Design Space

Overall, our design space had 191 parameters, nested as much as 4 levels deep. We now describe how we searched this space to building a customized solver. Identifying a set of parameters that optimize a given algorithm's performance on a given dataset is called *algorithm configuration*. There exist a wide variety of algorithm configuration tools [Ansótegui et al., 2009, Hutter et al., 2009, 2011, López-Ibáñez et al., 2016]. We used Sequential Model-based Algorithm Configuration (SMAC) [Hutter et al., 2011], the publicly available method that arguably achieves the best performance (see e.g., Hutter et al. [2014a]). SMAC uses the "Bayesian optimization" approach of interleaving random sampling and the exploration of algorithm designs that appear promising based on a learned model.

Unfortunately, even after performing algorithm configuration, it is rare to find a single algorithm that outperforms all others on instances of an NP-complete problem such as SAT. This inherent variability across solvers can be exploited by *algorithm portfolios* [Gomes and Selman, 2001, Nudelman et al., 2003, Xu et al., 2008]. Most straightforwardly, one selects a small set of algorithms with complementary performance on problems of interest and, when asked to solve a new instance, executes them in parallel. Of course, we wanted to construct such algo-

[5]The full algorithm description can be found in Section 4.4 of Fréchette et al. [2016].

rithm portfolios automatically as part of our deep optimization approach. We did this by using a method called `Hydra` [Xu et al., 2010] which runs iteratively, at each step directing the algorithm configurator to optimize marginal gains over the given portfolio. This allows `Hydra` to find algorithms that may perform poorly overall but that complement the existing portfolio. Overall, we ran `Hydra` for 8 steps, thereby producing a portfolio of novel solvers (dubbed SATFC) that could run on a standard 8-core workstation. The incentive auction used SATFC 2.3.1, which is available online at https://github.com/FCC/SATFC.

3.4 Data from Auction Simulations

During the development of SATFC, the FCC shared with us a wide range of anonymized problem instances that arose in auction simulations they performed in order to validate the auction design. These formed the "training set" we used in the deep optimization process when constructing SATFC 2.3.1.[6]

While SATFC 2.3.1 is itself open-source software, it is unfortunately impossible for us to share the data that was used to build it. We opted for what we hope is the next best thing: *evaluating* SATFC 2.3.1 and various alternatives using a publicly available test set. We thus wrote our own reverse auction simulator (which we will detail extensively in Chapter 4) and ran 20 simulations.[7] We sampled 10 000 "nontrivial" UHF problems uniformly at random from all of the problems across all simulations to use as our dataset, where we defined nontrivial problems as those that could not be solved by greedily augmenting the previous solution. Fewer than 3% of UHF problems in our simulations were nontrivial. This test set consisted of 9 482 feasible problems, 121 infeasible problems, and 397 problems that timed out at our 1 minute cutoff and therefore have unknown feasibility.

These problems are available as a benchmark (see Appendix A.14).

[6] These simulations explored a very narrow set of answers to the questions of which stations would participate and how bidders would interact with the auction mechanism; they do not represent a statement either by us or by the FCC about how these questions were resolved in the real auction. It is of course impossible to guarantee that variations in the assumptions would not have yielded computationally different problems.

[7] These simulations used the MCS value model (see Chapter 4.2.1), an 84 MHz clearing target, and a 60 second cutoff for SATFC 2.3.1.

3.5 Runtime Performance

We now evaluate SATFC's performance by contrasting it with various off-the-shelf alternatives. The FCC's initial investigations included modeling the station repacking problem as a mixed-integer program (MIP) and using off-the-shelf solvers paired with problem-specific speedups [FCC, 2014]. Unfortunately, the problem-specific elements of this solution were not publicly released, so we cannot evaluate them in this chapter. Instead, to assess the feasibility of a MIP approach, we ran what are arguably the two best-performing MIP solvers—`CPLEX` and `Gurobi`—on our test set of $10{,}000$ non-trivial instances. To encode the station repacking problem as a MIP, we created a variable $x_{s,c} \in \{0, 1\}$ for every station–channel pair, representing the proposition that station s is assigned to channel c. We imposed the constraints $\sum_{c \in D(s)} x_{s,c} = 1 \; \forall \; s \in S$ and $x_{s,c} + x_{s',c'} \leq 1 \; \forall \; \{(s, c), (s', c')\} \in I$, ensuring that each station is assigned to exactly one channel and that interference constraints are not violated. Both MIP solvers solved under half of the instances within our cutoff time of one minute; the results are shown in Figure 3.3. Such performance would likely have been insufficient for deployment in practice, since it implies unnecessarily high payments to many stations.

As already discussed, there exist a wide variety of SAT solvers that are available for use off the shelf. Figure 3.3 illustrates the performance of the 20 state-of-the-art solvers we considered in our initial configuration experiments in their default configurations. With few exceptions, the SAT solvers outperformed the MIP solvers, as can be seen by comparing the solid and dashed lines in Figure 3.3. However, runtimes and percentages of instances solved by the cutoff time were still not good enough for us to recommend deployment of any of these solvers in the actual auction. The best solver in its default configuration, `Gnovelty+PCL`, was able to solve the largest number of problems—79.96%—within the cutoff. (As mentioned earlier, the SATenstein design space includes `Gnovelty+PCL` alongside many other solvers.) The parallel portfolio of all 20 solvers from Figure 3.3 was little better, being able to solve only 81.58% of problems.

Lastly, we turn to SATFC 2.3.1. This 8-solver parallel portfolio stochastically dominated every individual solver that we considered and achieved very substantial gains after a few tenths of a second. It solved 87.73% of the problems in under a

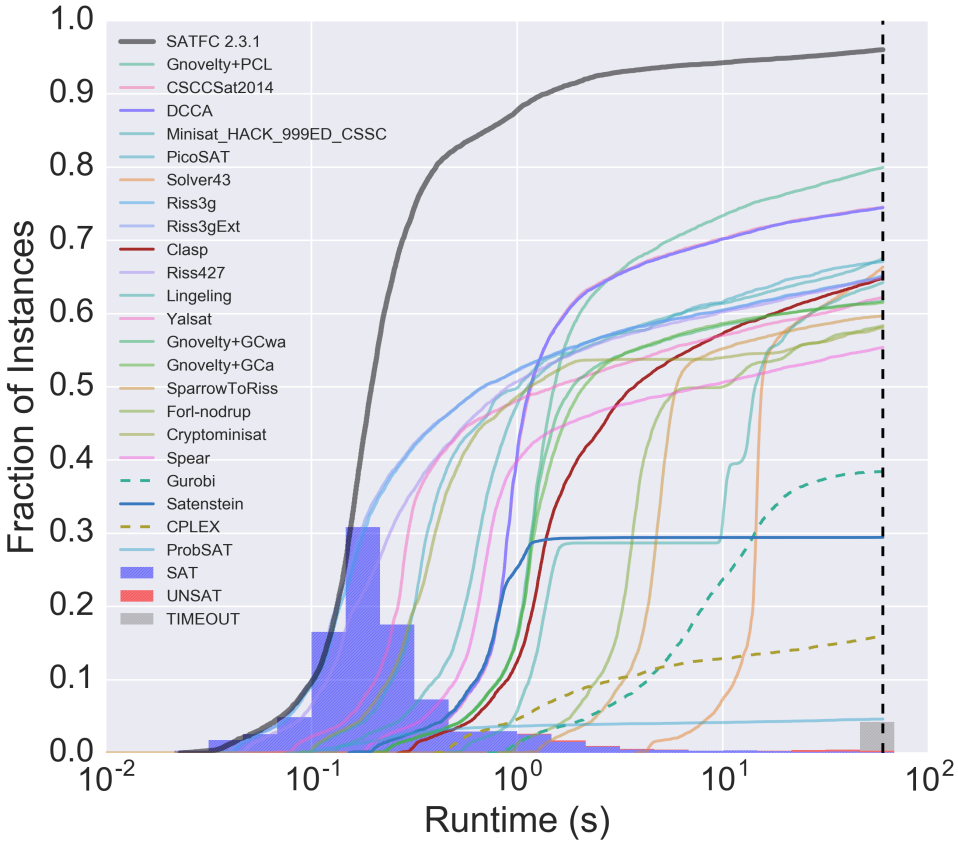


Figure 3.3: Empirical Cumulative Density Function (ECDF) of runtimes for default configurations of MIP and SAT solvers and for SATFC 2.3.1. The curves show fraction of instances solved (y axis) within different amounts of time (x axis; note the log scale). The legend is ordered by percentage of problems solved before the cutoff. The histogram indicates density of SAT and UNSAT instances binned by their (fastest) runtimes; unsatisfiable instances constituted fewer than 1% of solved instances.

second and 96.03% within the one-minute cutoff time. The histogram at the bottom of the figure indicates satisfiability status of instances solved by SATFC grouped by runtime; our instances were overwhelmingly satisfiable.

3.6 Conclusions

We designed a customized solution to the station repacking in the incentive auction problem using an approach we dub deep optimization. Specifically, we drew on a large parameterized design space to construct a strong portfolio of heuristic algorithms: SATFC 2.3.1, an open-source solver that was used in the real auction.

Chapter 4

Incentive Auction Design Alternatives: A Simulation Study

In this chapter, we switch gears from market operations to market analysis. We revisit the incentive auction, shifting the question from "how to make it work" to asking "how well did it work".

4.1 Introduction

The modern AI revolution has been fueled by computational processes whose performance improves as they are allocated more data and compute power. As computers have become faster and cheaper, and as data sets have grown larger, algorithmic approaches ranging from self-supervised neural networks to Monte Carlo Tree Search have exhibited increasing performance. Unfortunately, most mainstream market design approaches do not exhibit such scaling: purely theoretical analyses are unaffected by Moore's Law; rationality assumptions often make observations of actual bidder behavior irrelevant beyond the (worthwhile but narrow) question of estimating valuations via econometric techniques. How might we leverage cheap compute power and abundant data to complement the work of auction analysts? One possibility is running extensive simulations that leverage a high-fidelity market simulator and realistic models of participant preferences. A (mostly) recent literature has applied this approach across a diverse variety of domains, each

with their own literature, including matching mechanisms for refugee resettlement [Delacrétaz et al., 2023]; school assignment policies Allman et al. [2022]; dock allocation and rebalancing in bike sharing networks [Freund et al., 2018]; patrolling strategies for wildlife protection [Yang et al., 2014]; reallocating fishing licenses [Bichler et al., 2019]; and matching in organ exchanges [Santos et al., 2017].

This chapter develops and applies a qualitatively similar approach to the investigation of the incentive auction. There are many open questions in this setting: the auction's design was both novel and extremely complex [Leyton-Brown et al., 2017], and as a result it was not possible to thoroughly consider every potential design variation before the auction was run. Furthermore, the complexity of the setting presents a significant barrier to purely theoretical analyses. Our goal in this chapter is to show how a computationally intensive approach can be used to evaluate how well the auction design performed, particularly asking which elements of the design were most important and which variations of the design might have led to even better outcomes. Such insights can inform other important resource allocation problems that may leverage parts of the incentive auction's design. We speculate on one example: Just as historical US television broadcast rights varied by location and interference protections and were difficult to adapt to valuable new uses, historical rights to surface water in Western states have similar characteristics. Spectrum uses in a geographic area often leak interfering radiation into adjacent areas (a negative externality), and farm irrigation often leaks usable water back into river systems and acquifers (a positive externality). To develop a voluntary auction system that accounts for externalities and reorganizes water rights to unlock value, market designers will benefit from a deep understanding of which design details of the incentive auction contributed most to its successes and which could be simplified or improved (see, e.g., [Ferguson and Milgrom, 2023]). More broadly, we hope this work will serve as an example for how AI can be employed to understand and evaluate alternative market designs in complex settings.

4.1.1 Evaluating Complex Auction Designs

The design space for auctions is large [Anandalingam et al., 2005, Pekec and Rothkopf, 2003] and includes choices ranging from eligibility [Engelbrecht-Wiggans

and Kahn, 2005] and payment [Day and Raghavan, 2007] rules to bidding languages [Bichler et al., 2022].

To date, the design and analysis of auctions has relied primarily on theoretical tools. Such analysis becomes increasingly difficult as auctions become more complex,[1] e.g., as the number of goods at auction increases; as the number of parameters needed to describe valuation functions grows; as heterogeneity across bidders increases; etc. Nevertheless, the incentive auction's design was strongly informed by theoretical analysis, which has shown that deferred acceptance (DA) auctions have many good properties. DA auctions exhibit "unconditional winner privacy" [Milgrom and Segal, 2020], meaning that winners are only required to reveal as much about their valuation as is necessary to prove that they are winners. When stations' bids are binary responses to a series of descending offers and when stations are all independently owned, DA auctions are obviously strategy-proof [Li, 2017] and weakly group strategy-proof. When the sets of stations that can be jointly repacked form a uniform matroid, DA auctions repack the efficient set of stations [Bikhchandani et al., 2011]; if furthermore bidder values are drawn independently from known distributions, scoring offers according to virtual values can implement the Myerson "optimal auction" [Milgrom and Segal, 2020].

In exchange for the universality of its findings, theoretical analysis necessarily relies on simplifications, such as modeling the sets of stations with interference-free repackings as a uniform matroid. Such simplifications can raise questions about the practical applicability of results. For example, [Weiss et al., 2017, p. 51] warn that: "Deriving analytic results for [combinatorial auctions] is very challenging... insights from small, stylized models often do not translate to practical real-world problems".

A second approach for evaluating auction designs is data driven. There is a vast literature on laboratory and field experiments for auctions [Adomavicius et al., 2012, Cason et al., 2011, Ferejohn et al., 1979, Kwasnica et al., 2005, Rassenti et al., 1982]. To our knowledge no such experimental data exists evaluating the incentive auction; indeed, it would be very challenging to conduct realistic exper-

[1] For example, Kelly and Steinberg [2000] note of the Progressive Adaptive User Selection Environment (PAUSE) auction format that their design is "probably too complex to admit much theoretical analysis".

iments, given the real auction's long length and extremely large number of participants. Data can also be obtained from an auction's practical deployment. There is an extensive structural estimation literature in auctions and matching markets. Such analysis has led to significant insights in data-rich domains such as school choice [Agarwal and Somaini, 2018, Calsamiglia et al., 2020], highway procurement auctions [Krasnokutskaya and Seim, 2011, Somaini, 2020], timber auctions [Athey et al., 2011], and ad auctions [Athey and Nekipelov, 2010]. Unfortunately, only a single auction has been held to date using the incentive auction design, leaving us with only a single sample of past bids. We nevertheless leveraged this single datapoint to inform a valuation model, described below.

This chapter focuses on a third approach, computational simulations. This approach can verify whether theoretical findings from simplified models carry over to more complex, real-world settings and does not rely on the availability of rich historical data. Simulations are particularly well suited to studying market settings with complex clearing mechanisms (because such complexity tends to preclude "cleaner" analytical results) and uncontroversial models of agent behavior (because this reduces the risk that the model upon which the simulation relies will produce unrealistic behavior).

Other questions about the incentive auction's design were addressed via simulations. The FCC conducted its own (mostly unpublished) internal simulations, leveraging a variety of techniques from operations research [Kiddoo et al., 2019]. The broader research community conducted simulation studies of the reverse auction at various points in the design process and differing in the fidelity with which they modeled the auction mechanism; the degree to which they made simplifications for computational reasons; and the valuation models they employed. Before the auction mechanism was finalized, Kearns and Dworkin [2014] used simulations to characterize the space of feasible repackings based only on the interference constraints, e.g., estimating the relationship between the number of broadcasters relinquishing licenses and the feasibility of different clearing targets. This work used a bidder model that consisted entirely of determining which stations would participate, which they studied both using independent coin flips for every station and more complex models in which stations affiliated with the same broadcast networks made correlated decisions. Feasibility testing was performed using off-the-shelf

SAT solvers. Later in the design process, [Cramton et al., 2015, p. 31] used simulations to lobby for design changes such as changing the scoring rule and removing Dynamic Reserve Pricing (DRP). They leveraged a (non-public) valuation model developed through "discussions with many broadcasters, taking into account revenue data, historical station sales prices, station affiliation information, total market revenue, and other factors". DRP was ultimately scrapped but remains a topic of interest [Bazelon, 2022]. After the auction concluded, Doraszelski et al. [2017] used simulations to estimate how profitable and how risky bidder collusion strategies might have been, restricting experiments to regional scale and furthermore employing approximations ("limited repacking") to speed up computation. They concluded that the auction could have mitigated the harm imposed by collusion by restricting the participation of affiliated bidders in the auction. This work leveraged a novel value model for bidders, which we discuss further in Section 4.2.1 because we used it in our own simulations. The simulator used in this chapter more accurately simulates the actual reverse auction design than any other simulator of which we are aware, both in terms of its scale (national) and its coverage of the auction rules (e.g., multi-stage auctions including VHF bands).[2] Finally, Ausubel et al. [2017] performed a post-mortem analysis of the incentive auction in a similar vein to this chapter, but with a primary focus on the forward auction; we discuss one of their proposed amendments to the clearing algorithm in Section 4.4.2.

4.1.2 Our Simulation Methodology

We advocate and employed the following simulation methodology (see Figure 4.1):

1. Build an auction simulator (choosing an appropriate level of abstraction, as auction rules are often incredibly complex);

2. Create a bidder model, exposing parameters that control both valuations and behavior;

3. Establish a probability distribution over parameters of the bidder model;

[2] We make no comment about the fidelity of the FCC's own simulator, since details about it are not publicly available.

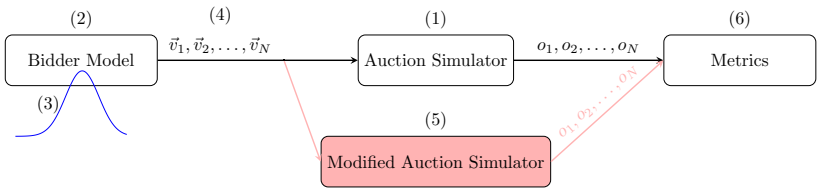


Figure 4.1: An overview of our simulation approach. N samples from a bidder model are fed into both a modified and unmodified simulator. The two outcome sets are converted into metrics and compared on a per-sample basis.

4. Identify many plausible auction scenarios by sampling repeatedly from this probability distribution;

5. Run paired simulations by holding this population of scenarios fixed and varying one or more elements of the auction design; and

6. Compare outcomes across paired samples using predetermined metrics.

For the study described in this chapter we instantiated these steps as follows. First, we custom-built the simulator used in this chapter. Our simulator's code is freely available online at http://www.cs.ubc.ca/labs/beta/Projects/SATFC/.

Next, perhaps the most critical step in our methodology is the specification of a bidder model, describing both valuations and behavior for each bidder. Since we focus on the reverse auction, each of our bidders corresponds to a television station. We considered two different valuation models: (1) the only fully specified model from the literature of which we are aware; and (2) a model we created based on publicly released bid data. We ran simulations with both models and contrast the results to investigate the robustness of our findings. Realistically modeling agent behavior is perhaps easiest in settings giving rise to *obviously dominant* strategies [Li, 2017]. A strategy is obviously dominant if any time an agent might deviate from the strategy, the best it can do under the deviation is no better than the worst it can do by following the strategy. Descending clock auctions make truthful bidding obviously dominant for bidders who own a single broadcast station (but not for those who own multiple stations).[3] Unfortunately, the real reverse auction lacked

[3] An agent can deviate from truthful bidding in this setting either by (1) accepting an offer below

obviously dominant strategies because stations could do more than just accepting and rejecting an offer: they could also accept a lower price and downgrade to a VHF channel. We decided to study the problem by considering the assumptions both that VHF bidding could be abstracted away and that it could not. For the latter case, no equilibrium bidding strategy is known, let alone any model of how bidders would behave if they believed that some or all of their opponents were bidding out of equilibrium. Given the widespread belief that many bidders in this auction were unsophisticated, we modeled bidders as bidding truthfully and myopically (see Section 4.2.2).

Finally, we restricted most of our comparisons to simulations that cleared the same amount of spectrum, considering two key metrics: the sum of values of stations removed from the airwaves (value loss) and the aggregate amount paid to stations (cost). When assessing design elements that did affect the amount of spectrum cleared, we assumed that it was preferable to clear more spectrum, based on statements made by the FCC about the auction's intended goals [FCC, 2012].

4.1.3 Questions Considered in Our Analysis

We ask four categories of questions about economic design and one about algorithmic design.

1. *How important was it to expand the set of products included in the incentive auction?* In the incentive auction, it would have been straightforward only to buy back licenses from UHF stations, since only UHF spectrum needed to be cleared. Instead—considerably increasing complexity—the FCC offered to purchase licenses from both VHF and UHF stations and offered UHF stations the option of moving into the VHF band rather than going off-air. This increased the pool of stations eligible to participate in the auction by roughly 20%, with the potential both to increase efficiency (a lower-value VHF station could go off-air to make room for a UHF station) and to lower

its value, or (2) by rejecting an offer above its value. Since prices only descend, these deviations either lead to selling at a loss or not selling at all, thus having a best-case utility of zero. Since truthful bidding has a worst-case utility of zero, it constitutes an obviously dominant strategy in this setting.

costs (c.f. Bulow and Klemperer's [1996] result that increasing competition can be more important to revenue than setting an optimal reserve price).

Was the extra complexity worth it? We found that it was (Section 4.4.1): under straightforward bidding, not repacking the VHF band could have increased the auction's cost by 20–30% and decreased efficiency by 5–10% (with variation depending both on random sampling and the choice of value model). Similar issues could arise in designing an auction for water rights: for example, one might ask whether the process should include ground water as well as surface water.

2. *What was the impact of the FCC's method of determining supply?* The literature has studied various auction designs in which the seller can adjust supply after observing demand [Back and Zender, 2001]. In the incentive auction, the FCC separately ran forward and reverse auctions at a given clearing target; then, if forward-auction demand was insufficient to cover reverse-auction costs, the clearing target was lowered and a new stage of the auction began. This design was novel and received little advance comment from stakeholders. We thus investigated how it performed, comparing it to a hypothetical auction having oracle access to the final clearing target. We found that the clearing procedure led to significantly higher costs and less efficient outcomes (Section 4.4.2): across all experiments, the clearing procedure increased average value loss by 5–26% and average cost by 4–50% (with variation depending on random sampling, the choice of value model used, and whether the VHF band was repacked). We then introduce a new simple clearing procedure that performs nearly as well without knowing the final clearing target.

3. *Was price discrimination effective at reducing payments to stations and increasing the number of channels cleared?* A canonical insight from revenue maximization [Myerson, 1981] is that bidders with weaker valuation distributions should be boosted to increase competition with stronger bidders. The incentive auction attempted to do something similar, reducing initial price offers for stations that reached smaller populations of viewers. The reduced price offers, called "pops scoring", were politically contentious. In

Section 4.4.3, we show that the effects of scoring were not robust across value models. Under our new value model, pops scoring led to average costs 5% (2%) higher than head-to-head pricing when the VHF band was (was not) repacked. Under the value model from the literature, head-to-head pricing led to average costs that were 39% (5%) higher relative to pops scoring when the VHF was (was not) repacked.

4. *How important was it to optimize a solver for the computationally hard problems embedded in the auction design?* Auctions can include problems that (in general) may not be solvable in a reasonable amount of time; a well-studied example is the winner determination problem in combinatorial auctions [Rothkopf et al., 1998, Sandholm, 2000]. Exact solutions can be replaced with approximations, but this may degrade outcomes and incentives. Harking back to the previous chapter, we asked: Was auction performance significantly improved by the FCC's use of a customized feasibility checker to determine whether a station could be repacked alongside the set of stations continuing over-the-air broadcasting? How large might that effect have been? The auction was robust from an incentive perspective to not solving every repacking problem, but the impact on cost and efficiency was harder to reason about. This question is important because the design of customized feasibility checkers required a nontrivial effort; such efforts should only be made in the future if they yield gains. We answer this question affirmatively in Section 4.4.4. We show that substituting the custom feasibility checker with the best off-the-shelf alternative could have increased both average costs and value loss by more than 20%.

5. *Were the reverse auction outcomes efficient?* A natural alternative auction format to compare the reverse clock auction to is the Vickrey-Clarke-Groves VCG auction. When bidders can have any values, the VCG auction is the only truthful mechanism that always selects efficient outcomes and entails no payments to losing bidders [Green and Laffont, 1979]. In the context of the incentive auction, running VCG would not have been possible. Finding the efficient repacking is computationally out-of-reach, and even very nearly approximately correct solutions could heavily distort incentives [Milgrom

and Segal, 2020]. To provide at least a partial answer to the question of the reverse auction's efficiency, we restricted ourselves to regional simulations and in Section 4.4.5 show that the reverse auction achieves nearly efficient outcomes.

The rest of the chapter proceeds as follows. Section 4.2 defines our valuation model and bidding model. Section 4.3 details our experimental setup and Section 4.4 reports our experimental findings. Appendix A contains additional results and technical material.

.

4.2 Value and Bidding Models

This section begins by describing two value models. The first follows Doraszelski et al. [2017]; we created the second for this study, based on bid data released after the auction by the FCC. An advantage of considering two different value models is that we were able to compare simulation results under both settings to assess the robustness of our findings. We conclude the section by describing a model of how stations bid as a function of these values.

4.2.1 Value Models

Each station s has a value $v_{s,b}$ for broadcasting in each permissible band b. We normalize so that a station has no value for being off-air, i.e., $v_{s,\text{OFF}} = 0$. Both models only provide $v_{s,\text{UHF}}$, that is, home band values for UHF stations. For the two VHF bands in the auction, lower and higher VHF, we model a UHF station's value for switching to the HVHF band as $\frac{2}{3} \cdot v_{s,\text{UHF}} \cdot \mathcal{N}(1, 0.05)$ and similarly for the LVHF band as: $\frac{1}{3} \cdot v_{s,\text{UHF}} \cdot \mathcal{N}(1, 0.05)$—i.e., roughly two thirds and one third of the station's UHF value with some multiplicative Gaussian noise. We describe robustness experiments on alternate parameter choices in Appendix A.8; our qualitative findings remain the same. We generated values for VHF stations by computing a hypothetical UHF value and then applying the fractional reductions for VHF bands just described.

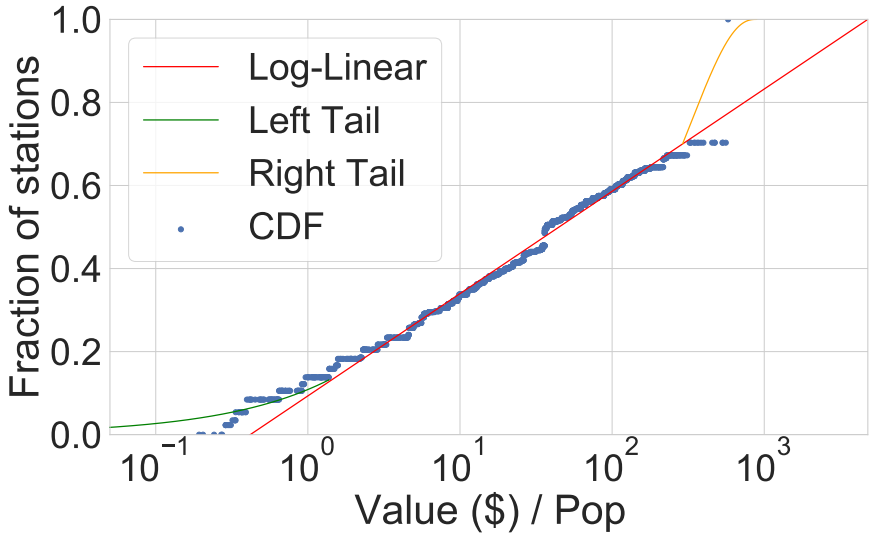


Figure 4.2: CDF for our maximum likelihood estimate of N and its log-uniform distribution fit, plus generalized Pareto left and right tails.

The MCS Value Model.

Doraszelski et al.'s [2017] valuation model, which we dub the MCS (Max of Cash flow and Stick value) model,[4] treats a station's value as the maximum of its cash flow value as a business and its stick value. The stick value represents the value of the broadcast license and tower, independent of the business; it can be more appropriate than cash flow when valuing non-commercial stations. Both of these were estimated from various sources including transaction data of station sales, advertising revenue, and station features.

A Novel Value Model based on Bid Data.

Two years after the incentive auction concluded, the FCC released the auction bids. We used this data to construct a "realistic" model for station valuations. While the bids are not sufficient to reveal station values, they do allow us to infer bounds on values. In some cases these bounds are relatively tight, with upper and lower

[4]Our implementation follows the value model described in a 2016 draft of the paper; the authors have subsequently made revisions (e.g., to the way that population is counted for low-power non-commercial stations).

bounds separated by a single clock interval. Most of the time they are looser, in some cases, to an extent that does not allow us to improve on the trivial bound.

We inferred bounds on each UHF station's home band value, $v_{s,\text{UHF}}$. For details, see Appendix A.3.1. We then used these bounds to fit a model. We assumed that value is proportional (in expectation) to population,[5] $v_{s,\text{UHF}} = \text{Population}\,(s) \cdot n_s$. Here n_s is some number in units of \$/pop, sampled from an unknown cumulative distribution function (CDF) N. Our upper and lower bounds on a given station's home band value can be translated into upper and lower bounds on a station's \$/pop by dividing by s's population. We computed a non-parametric maximum likelihood estimate of the distribution function N. Note that by definition $N(y_s) - N(x_s) = \Pr(x_s \leq n_s \leq y_s)$. Our goal was to maximize the product of these terms subject to constraints ensuring that N is a valid CDF. The results (Figure 4.2) suggest that N is a log-linear function. Data was sparsest in the tails of the distribution, especially in the right tail, so we replaced the log-linear segments in both tails with Generalized Pareto Distributions (GPDs). For more details, see Appendix A.3.2.

To generate UHF values, we multiply a station's population by a sample from the modeled distribution of N. In what follows, we refer to this model as the BD (bid data) model.

4.2.2 Bidding Model

We now describe our model of how stations bid. A station participated in our simulations if its opening price for going off-air exceeded its value for continuing to broadcast in its home band, i.e., if $P_{s;\text{OFF};\text{Open}} \geq v_{s,\text{pre}(s)}$. After excluding 64 non-mainland stations (see Appendix A.4), we considered 1813 stations eligible to participate in our simulations: 1407 UHF, 367 HVHF, and 39 LVHF.

Our two value models differ substantially in the participation rates that they predict. In the MCS model, station values tend to be low relative to opening prices, leading to very high participation rates. For example, considering 10,000 sampled

[5] A television station's value obviously depends on more than just the population it can reach. It is likely that other important features include the station's location (e.g., high-income versus low-income areas), and whether or not it is a commercial operation, for example. We opted for a simple model that only uses population, which the FCC used as a proxy for station value, acknowledging it will not capture all of the heterogeneity in station value.

value profiles of UHF stations, the mean number of participants was 1416. 1196 of these stations participated in every sample, and only 46 had less than an 80% chance of participating. In contrast, when running the same experiment with the BD model, no station participated in every sample and average participation rates were 60% (877 stations), closer to the 64% participation rate of UHF stations (930 stations) in the incentive auction. For reference, a total of 1030 stations (including VHF stations) actually did participate in the incentive auction [FCC, 2019a].

In auctions with UHF options only, in which bidders are able only to remain off-air or to exit the auction, a single station faces a strategic situation in which its utility is "obviously" maximized by remaining off-air if the price exceeds its value and by exiting otherwise—that is, by bidding myopically [Li, 2017]. The situation when VHF options are included is no longer obvious in this sense, but we continue to assume for simplicity that when bidding in round t, a station selects the offer that myopically maximizes its net profit, $\arg\max_{b \in B_{s,t}} P_{s;b;t} + v_{s;b}$. When fallback bids are required, we again assume that stations select the option that maximizes their net profit. We note that in the released bid data, only 52% (13%) of bids to move into LVHF (HVHF) were successful. Given that stations seeking to move to a VHF band faced a meaningful risk that their bid might fail to execute, some bidders might have benefited by bidding on VHF bands before it was straightforwardly optimal to do so. However, despite the potential drawbacks of straightforward VHF bidding, we are not aware of any behavioral rule that can be applied to all bidders that is arguably more realistic.

4.3 Simulator Design and Experimental Considerations

As the reverse auction is simplest to reason about when only the UHF band is repacked, we ran both simulations that only repacked the UHF band and simulations that also repacked the VHF band. This allowed us to investigate which of our results generalize across settings. We ran simulations on both value models described in Section 4.2. Unless otherwise stated, in every one of our experiments we took 50 samples per treatment. We gave feasibility checks 60 seconds to complete (as in the real auction, though of course we were unable to use exactly the

same hardware) unless otherwise stated.[6] We now explain how we compare simulated outcomes and discuss some simplifications our simulations make relative to the real auction process. We report additional elements of our experimental setup details in Appendix A.4.

4.3.1 Metrics

One goal for the auction is efficiency: for any given clearing target, to maximize the total value of the stations that remain on the air, or equivalently, to minimize the total value of the stations removed from the airwaves. We focus on the latter definition—*value lost* instead of *value preserved*—because it is unaffected by value estimates for large, highly valuable stations that do not participate in the auction. That is, *value preserved* includes the values of easy-to-repack stations, even those that do not participate in any interference constraints, and leads to efficiency estimates near 100% when few stations go off-air relative to the number that remain on air, even when the number of stations going off-air is large relative to the number required by an optimal solution.

We define the *value loss* of an auction outcome as $\sum_{s \in \mathcal{S}} v_{s,\text{pre}(s)} - v_{s,\text{post}(\gamma,s)}$. Ideally, our metric for allocative efficiency would be the ratio of the value loss of a simulation's final assignment, γ, relative to an assignment from an efficient repacking γ^* that minimizes the value loss for a given value profile, i.e., $\frac{\sum_{s \in \mathcal{S}} v_{s,\text{pre}(s)} - v_{s,\text{post}(\gamma,s)}}{\sum_{s \in \mathcal{S}} v_{s,\text{pre}(s)} - v_{s,\text{post}(\gamma^*,s)}}$. In general, however, γ^* is too difficult to compute, so we convert our absolute metric into a relative one by comparing the value loss between two simulations' final assignments (i.e., the ratio of value loss ratios, noticing that the denominators which depend on γ^* cancel out in this case).

Our second metric is the cost of reaching the specified clearing target: the prices paid to all winning stations, $\sum_{s \in S_{\text{winners}}} \mathcal{P}(s)$.

We use the terms "efficiency" and "cost" below as abbreviations that refer to

[6] We note a subtlety regarding runtime measurements. Since the feasibility checker uses a wall-time cutoff, there will be some degree of unavoidable noise (i.e., problems that require time very similar to the cutoff threshold). As a result, different auction trajectories starting from the same value profile can occasionally be caused by such measurement noise rather than a given design change being tested. This is difficult to control for, but the effect is random and averages out across samples. One alternative (which we did not employ) to enhance repeatability would have been to cache results (e.g., as described in Section 4.4 of Fréchette et al. [2016]) so that a given station repacking problem always produced a deterministic outcome.

value loss and the total payments made to broadcasters that go off-air or change bands. Outcomes with high efficiency (low value loss) and low cost are preferable. It is straightforward to compare two outcomes if they both clear the same amount of spectrum and one is both more efficient and cheaper; otherwise, any comparison requires a judgement call about how the two metrics should be traded off.

4.3.2 Impairments

The incentive auction's design requires it to begin from a feasible channel assignment for the non-participating stations. However, it may not always be possible to find a feasible repacking for the non-participating stations in the set of remaining channels $\overline{\mathcal{C}}$ induced by the initial clearing target $\overline{c}$. The FCC's rules therefore allowed a small number of stations to be assigned to channels *within* the spectrum that was otherwise resold (i.e., channels in $\mathcal{C} \setminus \overline{\mathcal{C}}$), even though doing so degrades the desirability of the mobile broadband licenses sold in the forward auction. Such stations are referred to as "impairing". There are two types of impairments: those caused by non-US stations that would be present even if every US station participated in the auction ("essential impairments") and those caused by non-participating US stations.

Despite our best efforts to make our simulations realistic, we could not replicate the optimization procedure the FCC used to determine the initial clearing target $\overline{c}$ and the set of impairing stations. This optimization relied on data that is not publicly available: the Inter-Service Interference ("ISIX") constraints that determine which geographic areas are impaired when a station is placed on channels to be resold. Without the ISIX data, we also could not replicate the analogous optimizations that the FCC conducted between auction stages.

Instead of determining the initial clearing target via an optimization, except when otherwise noted, we started our simulations at the 84 MHz clearing target (the clearing target at which the real incentive auction concluded) and ran auctions for only a single stage. We did this both because running multiple stages of the reverse auction is computationally expensive and because multi-stage simulations depend on additional assumptions about the forward auction. A channel uses 6 MHz of spectrum, so a clearing target of 84 MHz corresponds to clearing 14 chan-

nels; see Figure A.1 for more details on possible clearing targets. We do explore multi-stage auctions that begin from other clearing targets (including 126 MHz, the clearing target on which the incentive auction began) in Section 4.4.2. Our methodology for selecting which stations to impair is described in Appendix A.7. In our simulations, stations marked as impairing do not interfere with each other or with stations assigned to channels in $\overline{\mathcal{C}}$.

4.4 Experiments

Our experiments are divided into five categories, based on which element of the auction design they investigate: (1) repacking the VHF band in addition to UHF; (2) choosing the order in which stations are processed via a scoring function; (3) determining a clearing target by iterating between reverse and forward auction stages; (4) determining which stations to freeze by checking the feasibility of station repackings; and (5) assessing the allocative efficiency of the reverse auction.

4.4.1 Repacking the VHF Band

The incentive auction reduced only the number of UHF channels, but repacked stations in three bands: UHF, HVHF, and LVHF. Repacking the two VHF bands offered the potential for cost savings and efficiency gains, as UHF stations might have been willing to accept a smaller payment to move to a VHF channel instead of going off the air and this could have constituted a net gain, even taking into account the need to compensate VHF stations for going off-air to make space. An optimal repacking for a given value profile can only become weakly more efficient when the VHF bands are included as more configurations of stations become available.

However, adding extra bands to the reverse auction complicated an otherwise elegant design (see Section 3.2.3 and Appendix A.2.1). Stations no longer possessed obviously dominant strategies and might have benefited from reasoning about when to move up the ladder. Price calculations became more involved as each option had to be priced appropriately. The bidding language had to be augmented with fallback bids. Also, unlike in UHF-only settings where freezing is permanent, VHF stations can freeze and later unfreeze within the same stage if other stations move out of their home bands, complicating bid processing. All of

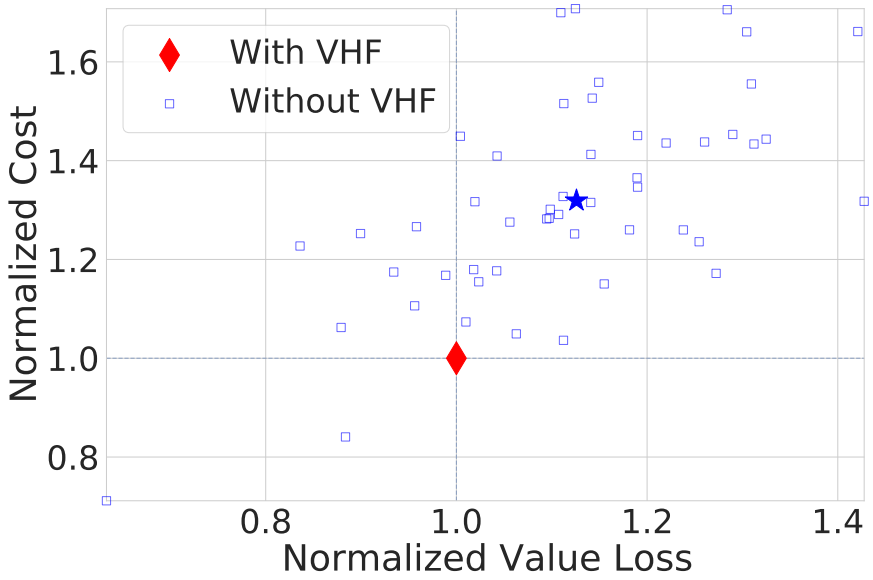


Figure 4.3: Comparing auctions that only repack the UHF band against auctions that also repack VHF bands.

this extra complexity made the auction more difficult to explain to station owners, which mattered since some of the participants were relatively unsophisticated and encouraging them to participate was a first-order concern. It is thus sensible to ask whether the additional complexity was worthwhile. While repacking VHF raised the possibility of more efficient allocations, such gains might have been small, arising only under exotic bidding behavior, or have come at a high cost. We thus asked: What changes to efficiency and cost arise when VHF options are included and bidders bid straightforwardly?

To answer this question, we ran two sets of auctions: the first repacking both the VHF and UHF bands, the other repacking only the UHF band. Our results are shown in Figure 4.3. (In what follows, we typically present figures for repacking both UHF and VHF using the BD value model, but discuss our findings across all experimental settings. The remaining graphs are presented in Appendix A.5.) Each point in the figure represents the outcome of one simulation, with its x-axis position denoting its efficiency and its y-axis position denoting its cost. Since it is difficult to show graphically which auctions use the same paired value profiles for

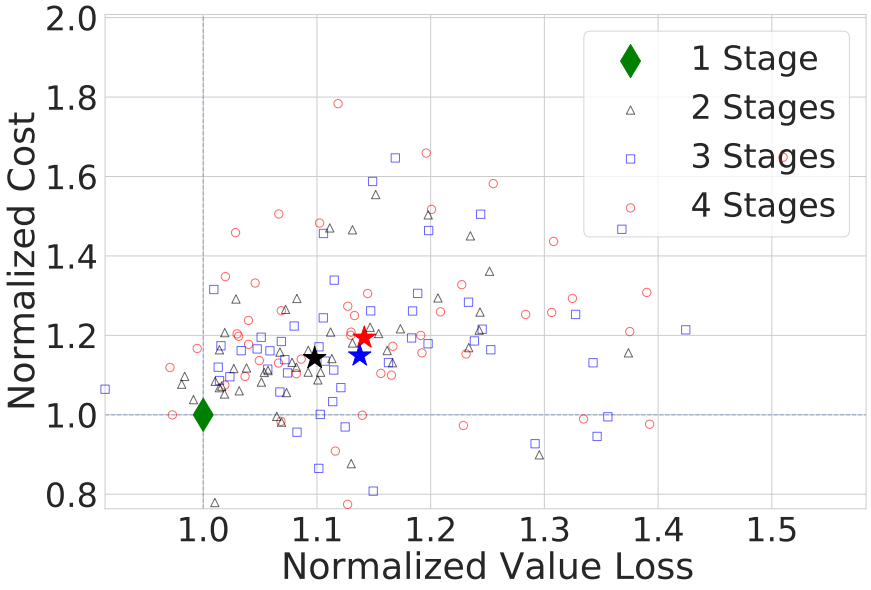


Figure 4.4: Comparing auctions running through 1-4 stages, ultimately ending on the same clearing target, with no impairing stations.

even a modest number of samples, rather than plotting raw efficiency and cost on each axis we instead plot normalized efficiency and cost. That is, we select one setting (in this case, auctions that repack VHF) as the reference treatment, and then plot the ratio of the efficiency (cost) of each simulation from additional treatments relative to the corresponding simulation using the same value profile in the reference treatment. With this choice, the reference treatment always corresponds to the point $(1, 1)$, represented in our figures by a diamond. For each treatment we also plot a star to indicate each metric's mean value.

The experiments just described took a little over one core-year to run. Using both value models, we observed that repacking the VHF band led to a significant reduction in payments—on average, sampled UHF-only auctions cost 1.23 and 1.32 times as much as their VHF-repacking counterparts using the MCS and BD value models, respectively. The impact on efficiency was more modest and less uniformly positive. Under the MCS (BD) value model, sampled UHF-only auctions experienced 1.05 (1.12) times higher value loss on average. On the whole, our simulations suggest that if bidders continued to bid straightforwardly despite

the complex design, repacking the VHF band was an important design choice that likely led to lower costs and also somewhat more efficient outcomes.

4.4.2 Multi-Stage Clearing

The incentive auction allowed for multiple stages of reverse followed by forward auctions in order to let market forces determine the appropriate amount of spectrum to clear. Each successive stage began with a reverse auction clearing a smaller amount of spectrum to achieve a lower overall cost than the previous stage. The actual incentive auction went through four stages.

One obvious practical drawback of a multi-stage approach is its impact on auction length: running multiple stages takes time. As noted by [Ausubel et al., 2017, p. 14] "One potential criticism of the incentive auction is that it lasted too long. If, after the initial commitment, the FCC had selected a clearing target of 84 MHz (instead of 126 MHz), it is very likely that the auction would have concluded after a single stage with significantly fewer rounds."

A less obvious concern is how the multi-stage approach impacted the final outcome. In general, if the clock auction perfectly optimizes for each clearing target and if the stations packed for the higher target are not a subset of those packed for the lower target, then one should expect a multi-stage auction to perform worse than an auction that starts by setting the correct target. We investigate these dynamics in Appendix A.9, considering elaborations of our previous worked example. First, we show how stations packed in late rounds of an early stage can cause problems in subsequent stages. Then, we show the clock auction's greedy optimization procedure can also lead to the reverse: an auctioneer *benefiting* from running a multi-stage rather than a single-stage auction.

Having observed that multi-stage clearing can yield both economic benefits and harms, we turn to simulations, asking two questions: (1) what, if any, economic costs arose due to multi-stage clearing? (2) Would an "early-stopping" alternative to the reverse auction have performed better?

The Economic Impact of Multi-Stage Clearing.

To assess the economic impact of multi-stage clearing, we ran experiments that began trying to clear 126, 114, 108, and 84 MHz of spectrum, each proceeding to follow the ordering of clearing targets selected by the FCC (see Figure A.1), and each terminating at 84 MHz, leading to four-, three-, two- and single-stage auctions, respectively. These experiments took 31 core-years.

A complicating factor in this comparison is that for any given value profile, the final set of impairing stations may differ based on the starting stage, which muddies the interpretation of cleared spectrum as a measure of performance. Our impairment mechanism (described in Appendix A.7) attempts to remove impairments in between stages. The cleared spectrum is only comparable if exactly the same set of impairing stations remain at the auction's termination. To improve the comparison, we analyze the results of these experiments in two ways. The first is to consider a world without impairments, corresponding to the case where the FCC is willing in principle to pay any amount to stations to eliminate impairments and therefore sets high opening prices. Under this assumption we observed that running the auction through multiple stages degraded both cost and efficiency, especially when the VHF band was repacked (Figure 4.4). In VHF-repacking simulations using the MCS (BD) model, on average four-stage auctions cost 1.50 (1.19) times as much as their single-stage counterparts and had 1.26 (1.14) times the value loss. The results were similar but less dramatic in magnitude for UHF-only auctions. In UHF-only simulations using the MCS (BD) model, four-stage auctions cost roughly 5% (10%) more and had roughly 5% (10%) additional value loss compared to perfectly forecasting the clearing target. In all cases, even if the exact stage was not perfectly selected, starting the auction closer to the final stage would also have yielded significant improvements to both metrics on average.

A second way of analyzing the results is to account for impairing stations. Of course, this leads to "apples-to-oranges" comparisons in the sense that the quality of cleared spectrum varies across simulations. We do not have a reliable way of assessing how much a given impairment devalues spectrum. The answer surely depends on the location and power of the exact stations in question, but a blunt proxy number for how bad a set of impairing stations might be is the sum of their

population (less is better). In our experiments, there were never any impairing stations remaining at the final stage when using the MCS value model, so results remained unchanged for this model. We visualize results for the BD model in Figure A.4. We observe much higher variance than in the previous analysis. With VHF repacking, we again observe that multi-stage auctions performed much worse than single-stage auctions, with four-stage simulations costing 1.25 times more and experiencing 1.10 times more value loss than single-stage auctions. In the UHF-only case we no longer observe the trend that more stages led to worse outcomes—on average, four-stage simulations cost and had value loss about 1% higher than single-stage simulations, and two- and three-stage simulations performed a little better (1–2%) on average on each metric. However, single-stage auctions resulted on average in the least impaired spectrum (according to our population metric): four-stage, three-stage, and two-stage simulations had mean impairing populations of 206, 205, and 206 million respectively compared to 192 million for single-stage simulations. Our results for repacking the VHF band look similar, with 205, 203, and 207 million mean impairing population for four-, three-, and two-stage auctions respectively compared to 192 million for single-stage auctions.

Early Stopping

The results just presented led us to ask: is there any other way the FCC could have determined a clearing target endogenously with less impact on cost and efficiency? We propose a simple answer, which we call *early stopping*: for each candidate clearing target, conduct the forward auction before the corresponding reverse auction, and stop each reverse auction as soon as its cost exceeds the forward auction revenue. To see why this would help, consider again our example in Figure 3.2, but now imagine that the auctioneer knows going into the first reverse auction stage that the forward auction revenue is some number less than V_A. Once station B freezes at a price of V_A, the provisional cost exceeds the forward auction revenue. Following our proposal, the first reverse auction stage would immediately stop and so station C would not exit. Then, in the second stage, station B would exit next and the outcome would exactly match that of an auction in which the clearing target had initially been set to the second stage target. In general, continuing the reverse

auction instead of aborting can only result in more stations exiting, limiting flexibility in later stages.[7] The original design proposed that the forward and reverse auctions be run in parallel, but due to finite staffing resources, the auctions had to be sequenced. We note that early stopping represents a potential algorithmic improvement even over the original parallel design.

Early stopping does not always outperform the original design; it is possible to construct examples (see Appendix A.10) where the commitments made in earlier stages are better than those made in later stages. Nevertheless, we believed that early stopping would typically help in practice. To test this, we ran two sets of experiments. The first set compared early stopping auctions against single-stage stations that "knew" the correct clearing target. The second set compared the amount of spectrum cleared by early stopping auctions to auctions using the original design. Both of these experiments required forward auction revenues as inputs. We had little data to use for modeling these revenues—one observation for each of the four clearing targets that were reached in practice—so we adopted a convenient model described in Appendix A.12. Unlike previous experiments where we assumed a predetermined number of stages, we now determined the auction's end by comparing forward auction revenues to clearing costs. This meant that auctions could end at any stage, so the amount of spectrum sold could differ across paired simulations. We assume that clearing more spectrum is preferable to clearing less, noting the FCC's stated goals. We ran early stopping auctions with an initial clearing target of 126 MHz (corresponding to the first stage of the real auction) and following the 600 MHz band plan (see Figure A.1) from that point on.[8]

[7] A suggestion along similar lines was made in Ausubel et al. [2017]. Rather than swapping the order of the reverse and forward auctions, they proposed forecasting reasonable bounds on forward auction revenue (i.e., asserting that it would be unlikely for telecoms to pay more than $X for a given amount of spectrum) for each stage. The reverse auction would then terminate when the provisional cost reached $X, the following forward auction would be skipped, and the next stage of the reverse auction would be begin. They argue that first, this would have reduced the auction duration, and second, that bidders in the forward auction had a sense in early stages that the prices were too high for the auction to terminate, so they did not bid sincerely.

[8] We note an implementation detail: The reverse auction detects "unconstrained" stations and forces them to exit at the end of each round. Unconstrained stations are those that can be provably feasibly repacked for the remainder of the stage, and hence would otherwise simply bid until their clocks wound down to zero or they exited of their own accord. In an early stopping auction, forcing these stations to exit no longer makes sense, as such stations may not be unconstrained in the next stage. Therefore, we disabled these checks in our early stopping simulations.

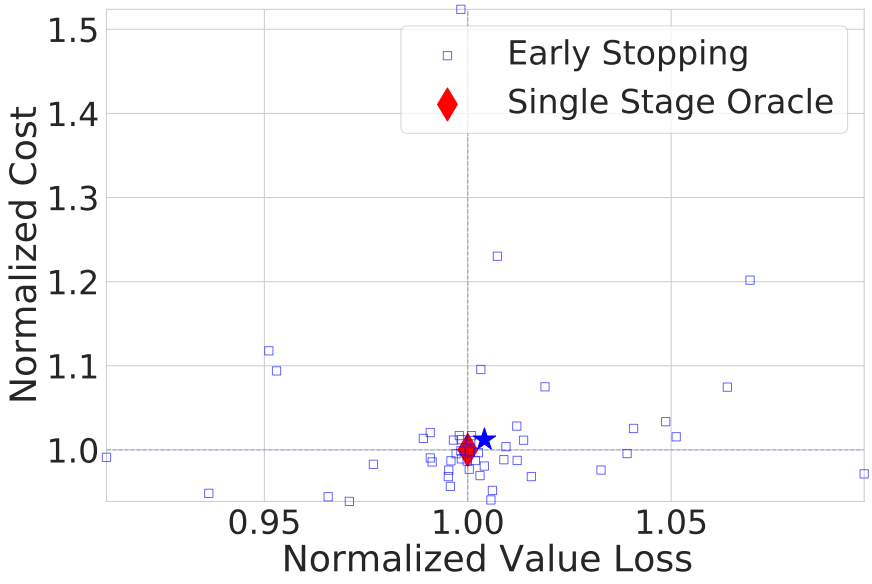


Figure 4.5: Comparing early stopping auctions against single-stage auctions.

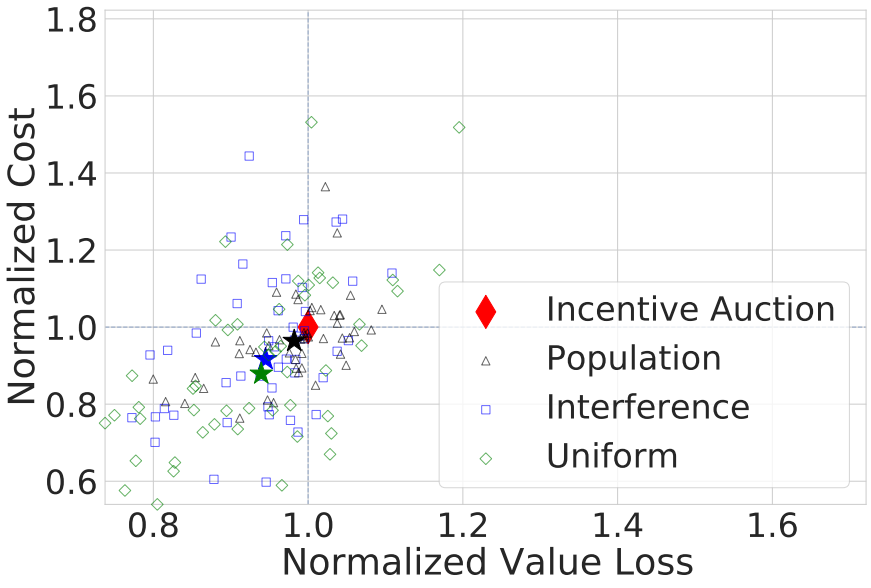


Figure 4.6: Comparing auctions using four different scoring rules.

Once these experiments completed, we determined the final stage of each early stopping simulation and ran a corresponding single-stage auction in order to assess the "penalty" due to early stopping vs. perfect forecasting of the clearing target. The results are shown in Figure 4.5. For computational reasons, we only ran UHF-only simulations; even so, the experiments took more than 15 core-years overall. To avoid making comparisons between different qualities of spectrum, we disabled our impairment mechanism and let prices start as high as required for full participation. We observed that in both value models, early stopping performed well relative to single-stage auctions, with increases to average cost and value loss of no more than about 1%. These results are particularly encouraging when compared to the multi-stage experiments described earlier, in which we observed significant gaps between multi-stage clearing and perfect forecasting.

We next ran simulations that compared early stopping to the original clearing procedure to determine which cleared more spectrum. Under both value models, on average more mobile licenses were created when by early stopping than the original algorithm—0.50 and 0.22 extra licenses under the BD and MCS models, respectively. Early stopping also led to shorter auctions, averaging 30% and 76% of the rounds required by the original algorithm under BD and MCS, respectively.

We conclude by reflecting on some concrete numbers from the real auction. At the end of the first round of the first stage, payments to frozen stations already exceeded $50 billion; when the stage finished, these payments were $86 billion. In the subsequent forward auction, revenues were only $23 billion. Early stopping would have terminated the reverse auction during the very first round of bidding. The first reverse auction stage took one full month to resolve. This month, and possibly more time in future stages, would have been saved if early stopping had been implemented.[9]

[9]One advantage of the ultimately implemented reverse-first ordering is that it allowed the forward auction to terminate early when it was clear from looking at the high-value products (Category 1 products in High Demand Partial Economic Areas) that the auction would require another stage, skipping the rounds that would have otherwise been required to let every product's demand settle. The speedups from terminating the reverse auction early would need to be traded off against terminating the forward auction early; we did not perform such an analysis.

4.4.3 Scoring Rules

Stations' starting prices in the incentive auction were not all the same: they were set proportionally to an assigned *score*, determined by a *scoring rule*. Under truthful bidding in a UHF-only auction, an unfrozen station will remain off-air for exactly as many rounds as it takes for its price to fall below its value, at which point it will exit. Using scoring to raise or lower a station's initial price relative to others is one control knob an auctioneer has over the order in which stations exit.

Theoretically, scoring performs two distinct functions. First, because every descending clock auction is equivalent to a greedy algorithm for packing stations into the broadcast spectrum, it may be possible to pack a larger and more valuable set of stations if the algorithm prioritizes stations that interfere with fewer neighbours. In the FCC's design, this was achieved by offering higher prices to stations with more "interference links"[10] so that those stations would be less likely to exit. Second, scoring reduces an auction's expected cost by offering lower prices to stations that would be likelier to accept them, following Myerson [1981]. In the FCC's design, this was implemented by reducing prices offered to stations serving smaller populations. (This design element was controversial: it was vigorously opposed by a coalition of owners of lower powered stations serving smaller populations, the "Equal Opportunity for Broadcasters Coalition", whose starting prices in the clock auction were reduced.) Overall, the two elements were combined by setting opening prices in proportion to the square root of the product of a station's population and its interference links.

In our experiments, we compared the following four scoring rules: (1) "Incentive Auction", the scoring rule used in the actual auction; (2) "Interference", the square root of a station's interference links; (3) "Population", the square root of a station's population; and (4) "Uniform", scoring each station identically. As in the actual auction, we normalized all scores so that the highest scoring UHF station had a score of 1 million.

We note a subtlety in these experiments: scoring rules impact prices, so simulations differing only in their scoring rules will not necessarily select the same set of impairing stations given the same value profile. To prevent "apples-to-oranges"

[10] More precisely: "an index of the number and significance of co- and adjacent channel interference constraints that station would impose on repacking." [FCC, 2015a]

comparisons across auctions clearing spectrum of varying quality, we disabled our impairment mechanism and ran simulations with high enough base clock prices[11] to elicit full UHF-station participation (thereby avoiding all "unessential" impairments).[12] Running with no impairment mechanism corresponds to considering a world in which the FCC is unwilling to accept any degradation to the cleared spectrum.

Results for our simulations using the various scoring rules are shown in Figure 4.6. Under the MCS value model, when the VHF band was repacked, we observed that simulations using only interference scoring cost 1.24 times as much and experienced 1.11 times as much value loss as simulations that combined population and interference scoring (the FCC's scoring rule). Uniformly scoring stations was not effective in this setting, with simulations averaging nearly 50% higher costs and 25% higher value loss. When only the UHF band was repacked, the scoring rule appeared to matter much less. Here, the interference scoring rule outperformed other scoring rules on average, with mean costs and value losses of 0.93 and 0.99 times respectively compared to corresponding simulations using the FCC's scoring rule. Under the BD value model, we observed much higher variance across simulations. If we nevertheless consider average performance, the FCC's scoring rule was outperformed in both metrics by every other scoring rule considered regardless of whether the VHF band was repacked. When the VHF band was repacked, the lowest value losses and costs were achieved, surprisingly, by the uniform scoring rule (94% mean value loss and 88% cost relative to the FCC's scoring rule). When only the UHF band was repacked, the lowest average value losses and costs were achieved by interference scoring (97% and 95% of the FCC scoring rule). On the whole, beyond a single setting (MCS values and UHF + VHF repacking) we did not find robust evidence for population-based scoring. These experiments required 3.5 core-years to run.

All of the scoring rules that we considered are static: they assign each sta-

[11] We continued to lock the VHF band until the base clock price reached the FCC's starting base clock price as described in Appendix A.7, because the heuristics that set VHF prices are fragile and we were concerned that altering them might cause unintended effects on the VHF allocation.

[12] Another reason not to run our impairment mechanism in this case is that the starting base clock price would surely change under a shift in the distribution of station scores; by not running our impairment procedure our results are not sensitive to this parameter.

tion a single, fixed score. The theory of descending clock auctions also considers scores that respond to auction history in a dynamic fashion (provided that they never increase a station's price). We leave the investigation of dynamic scoring rules for this setting as future work, but note that our simulations' failure to crisply recommend any of the static scoring rules we considered may motivate such an investigation.

4.4.4 Feasibility Checking

The feasibility checker determines if a given station can be repacked alongside the stations that have previously exited the auction. Feasibility checking was a large concern in the incentive auction because the station repacking problem is hard both theoretically—it is NP-complete—and in practice. When the feasibility checker cannot find a way to repack a station, either by proving that no repacking exists or by running out of time, a station freezes: it stops bidding and its compensation stops falling. The feasibility checker's quality therefore has a direct effect on both the cost and efficiency of the auction: if the feasibility checker fails to find an assignment that repacks a station when such an assignment exists, this station will freeze at an unnecessarily high price. If the station would otherwise never freeze at all, the value loss of the allocation is changed.

Do auctions achieve better efficiency and/or lower costs as the feasibility checker improves? Our intuition is that better feasibility checking should lead to better results. However, this intuition is not always correct; see Appendix A.11 for a counterexample.

Effect of the Feasibility Checker on Auction Outcomes.

SATFC 2.3.1, the feasibility checker that was used in the incentive auction, was designed over several years [Fréchette et al., 2016, Newman et al., 2017] as described in Chapter 3.[13] We now investigate whether the effort invested in making SATFC 2.3.1 was helpful, or whether a more off-the-shelf solution would have sufficed. To do so, we ran simulations in which we exchanged SATFC 2.3.1 with alternative

[13]While the prototype of SATFC was developed quickly, various unrelated legal and logistical delays to the auction's launch provided SATFC's authors with time to refine it.

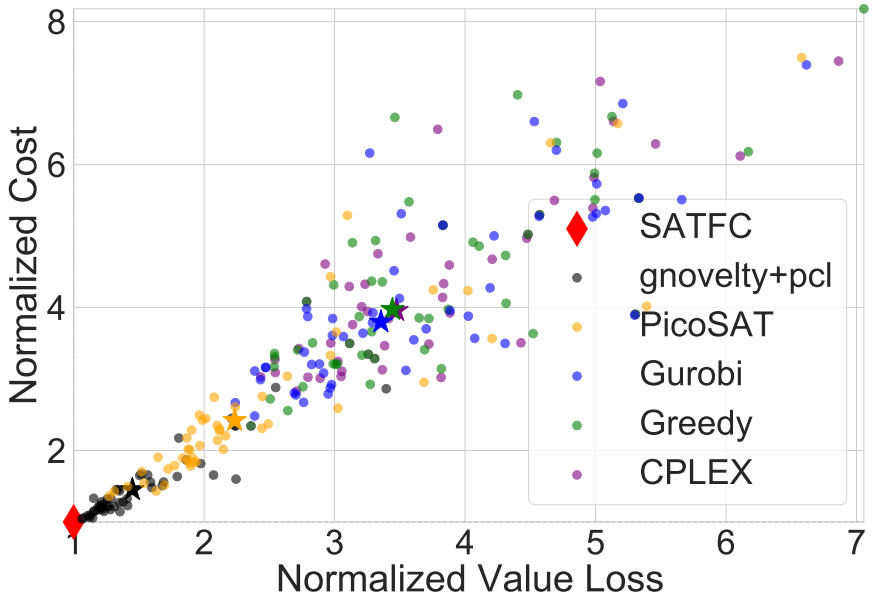


Figure 4.7: Comparing auctions using different feasibility checkers.

solvers.

Specifically, based on our analysis in the previous chapter (see Figure 3.3) we compared the following solvers: *SATFC 2.3.1*: the feasibility checker used in the incentive auction; *Greedy*: a solver that simply checks whether a previous assignment can be augmented directly, without changing the assignments of any other stations (this algorithm is the simplest reasonable feasibility checker and thus serves as a baseline); *PicoSAT*: To our knowledge, alongside MIP approaches, the only other solver that has been used in publications on the incentive auction, probably because it was shown to be the best among a set of alternatives in an early talk at the FCC on the subject [Leyton-Brown, 2013]; *Gurobi* and *CPLEX*: MIP solvers initially considered by the FCC; and *Gnovelty+PCL*: the best performing of the 22 ACLib solvers on the benchmark data described above.

Our experiments took just over two core-years. Results are shown in Figure 4.7.[14] We observed that stronger feasibility checkers led to better outcomes

[14]Our simulator always attempted to solve each feasibility check using the greedy algorithm before calling another solver, so every other feasibility checker can be understood as a sequential portfolio of the greedy solver and itself. We used this heuristic for good reason: the vast majority of feasibility

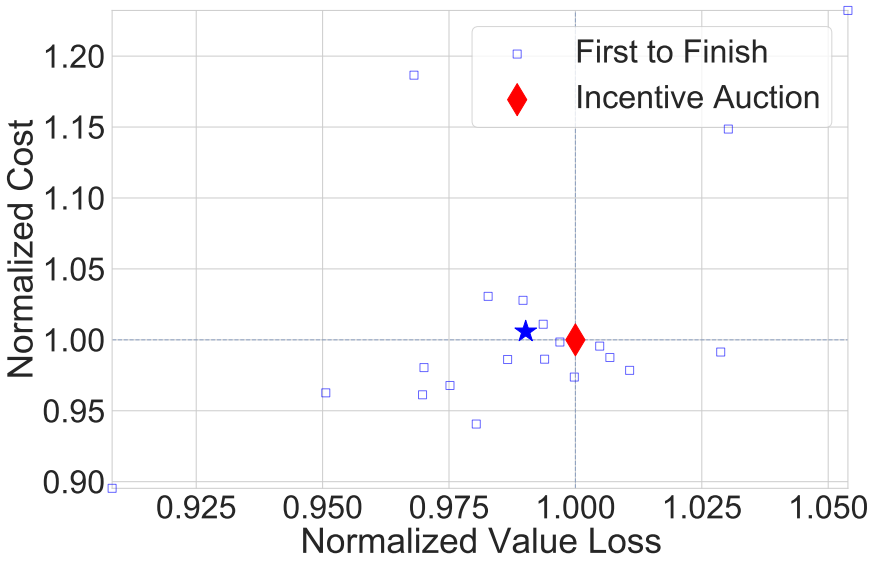


Figure 4.8: Comparing the first to finish algorithm against the standard bid processing algorithm for single-stage 126 MHz auctions.

according to both of our metrics: the relative rankings of the solvers in the benchmark study translated exactly into the relative rankings across both of our metrics, regardless of whether the VHF band was repacked and regardless of our choice of value model. In particular, we observed that SATFC 2.3.1 dominated all other solvers on both metrics, not only on average but in each individual simulation and across both value models.

Reverse auctions run using the best off-the-shelf solver, *gnovelty+pcl*, cost between 1.22 and 1.45 times more on average (depending on bands being repacked and the value model used) and lost between 1.22 and 1.45 times as much broadcaster value as those based on SATFC 2.3.1.[15]

checking problems encountered in a typical simulation can be solved greedily. Lastly, we note that for consistency we always used SATFC 2.3.1 as the feasibility checker in all experiments during the impairment phase (see Appendix A.7).

[15]We emphasize that these experiments considered only each solver's default parameter setting, and do not investigate the performance each solver could have achieved if it had been tuned or otherwise customized.

Alternative Bid Processing Algorithms.

Given our findings about the impact of a strong feasibility checker, we asked: Could allowing more time for solving individual repacking problems have led to fewer stations freezing unnecessarily and thereby have improved economic outcomes? Figure 3.3 makes a compelling case that problems that are not solved quickly are overwhelmingly infeasible, yet it is difficult to say without experimentation what the impact of solving the remaining truly feasible problems might be. Significantly increasing the cutoff time given to each problem would not have been practical. The first round of the reverse auction had 1030 bidding stations. A worst-case scenario of each sequential repacking problem taking one hour, say, would have required more than 42 days. This would have slowed the pace of the auction to a halt!

Going beyond feasibility checker comparisons, we explored two variants to the bid processing algorithm that increase the computation time available to processing stations before declaring them frozen while also respecting the time constraints of the auction. The first optimistically does not freeze a station with an indeterminate check, allowing future rounds to revisit whether the station is indeed frozen; the second leverages parallel computation to make better use of the full time window allotted to a given round.

Revisiting Indeterminate Feasibility Checks. The incentive auction froze a station whenever the feasibility checker could not prove that the station could be repacked. We note that indeterminate cases did not have to be treated this way: incentive constraints require only that a station's winning price does not decrease if it cannot exit. While calling these stations infeasible saves computation time as there is no need to recheck them in later rounds, it is possible that as other stations exit, a given station's repacking problem becomes more constrained and thus easier to solve. An alternative is for the bid processing algorithm, upon encountering an indeterminate feasibility check for a station, simply not to process that station's bid. Such a station is not frozen and will be asked to bid again in the next round. The prices offered to that station decreases as normal, but its winning payment $\mathcal{P}(s)$ will not decrease unless the solver finds a feasible repacking for the station. The

auction designers rejected this alternative on the grounds that the actual rules would be simpler for bidders, but until now the alternative has never been quantitatively investigated.

Figure A.8 compares the results of the standard bid processing algorithm against one that does not freeze stations with indeterminate results. These experiments took 3 core-years to run. We observed on average, in UHF-only simulations, small (about 0.5–1%) improvements to both metrics when we revisited indeterminate results.

The First to Finish Algorithm. Bid processing for each round was required to complete within a fixed time window. Since stations were checked sequentially, with a fixed cutoff time allotted to each check, and since empirically most checks finished very quickly (see Figure 3.3), many rounds terminated earlier than they were required to. Recognizing that stations' incentives are unaffected by the order in which their feasibility checks are performed, we propose an alternate bid processing algorithm (dubbed "first to finish") that aims to order feasibility checks in ascending order of their runtimes, ensuring that a round never terminates early unless all feasibility problems have been solved.[16] This can be achieved by running all feasibility checks in parallel, each with a cutoff equal to the entire amount of time remaining in the round. As soon as any feasibility check completes, we process the corresponding station's bid. If the station remains in its current band, we leave all other feasibility checks running unchanged; otherwise, we update them to include the changes implied by the just-processed station, and restart all of them in parallel with a new cutoff corresponding to all of the remaining time. We provide pseudocode in Algorithm 4.

The first-to-finish algorithm consumes dramatically more compute power than the standard bid processing algorithm: checking the status of all 1030 participating stations in parallel would require on the order of 10,000 CPUs, since each run of SATFC 2.3.1 uses 8 parallel threads. At the scale of the incentive auction

[16] In fact, we first raised this idea as the last details of the incentive auction were being finalized. Although the design team received the idea positively, it was not pursued because altering the bid processing algorithm would have contradicted the published auction rules, which was no longer possible by that time.

and given modern cloud computing resources, such computational requirements would not have been prohibitive. However, given that most feasibility checks are extremely fast (see Figure 3.3), a more practical variant might be to run each check sequentially with a tiny cutoff before switching to fully parallel. In our experiments (described below), in all rounds at most 27 stations had checks that took longer than 1 second, meaning this variant would have required dramatically fewer CPUs than suggested by the worst-case bound.

Our experiments compare the first-to-finish algorithm[17] with a one-hour-per-round cutoff against the traditional bid processing algorithm with the usual one-minute-per-problem cutoff. Due to the computational requirements of these experiments, we considered only 20 samples of UHF-only simulations; still, they took more than 6 core-years to run. The results are shown in Figure 4.8. We observed moderate noise in both metrics and a small average effect: about a 1% improvement in both metrics under the MCS value model, and a 1% improvement to value loss and a 0.5% increase in cost under the BD value model. We conclude that increased cutoffs would not have made a significant difference to the auction outcome. We do expect that the first-to-finish algorithm would have yielded larger gains if paired with a weaker feasibility checker or a larger incentive auction that gave rise to even harder feasibility checking problems.

4.4.5 Comparison to VCG in Restricted Setting

In this section we compare the FCC's reverse auction mechanism to a VCG mechanism. We chose to compare to VCG because it computes a globally efficient outcome and is truthful for bidders. VCG pays each winning station s the difference between the sum of values of stations other than s for γ^* and the sum of the same stations' values for a packing that is optimal subject to the constraint that s does not win. We identified these optimal packings using the MIP encoding from Section 3.5 with two changes. First, we added the objective of maximizing the

[17] Since we lacked access to the required number of parallel computers, we ran a sequential version of the first to finish algorithm. The sequential version runs all of the unprocessed checks for a small cutoff and then processes them in order of completion time until encountering a movement or exit bid, at which point checks are restarted. If at the end of the cutoff, only indeterminate problems remain, the cutoff is set to the minimum of double the current cutoff or the remaining time left in the round.

aggregate values of the participating stations:

$$\text{maximize} \sum_{s \in S_{\text{participating}}} \sum_{c \in D(s)} x_{s,c} \cdot v_{s,\text{band}(c)} \tag{4.1}$$

Here, band (c) is a function that returns the corresponding band for a given channel. Second, we allowed the option of not assigning a channel to a bidding station. Our formulation is detailed in Appendix A.13. In addition to efficiency, we are also interested in how VCG prices compare to prices in the FCC's mechanism. The $|S_{\text{winners}}|$ pricing problems can be solved in parallel once the initial allocation has been found.

Unfortunately, it was impossible to solve these optimization problems at a national scale, even given several days of computing time. Approximate results are not useful in this setting because even small approximation errors can lead to significant over-payments to stations [Milgrom and Segal, 2020]. We therefore constructed tractable problems by restricting ourselves to stations in the vicinity of New York City, one of the most densely connected regions in the interference graph. More specifically, we dropped all Canadian stations and restricted ourselves to the UHF band[18] using the largest possible clearing target of 126 MHz (corresponding to 16 channels available for repacking). Using the interference graph induced by these restrictions, we then dropped every station whose shortest path length to a station in New York City exceeded two. The result was a setting with 218 stations including those in Boston, Philadelphia and Washington DC. The setting yielded a MIP encoding with 2 465 variables and 78,717 constraints.

Using the MCS value model[19], we ran a VCG auction. We computed allocations and payments using *CPLEX*, solving all MIPs optimally to within 10^{-6} absolute MIP gap tolerance. We also ran reverse auction simulations using SATFC 2.3.1. In total, our simulations consumed over 5 core-years (dominated by the VCG simulations).

Figure 4.10 illustrates the results. The SATFC simulations had a mean value loss ratio of 1.048 and a mean cost ratio of 0.760, indicating that the reverse auction

[18] While we hoped to extend these experiments into a VHF setting, we were unable to solve problems even after running for several weeks.

[19] These experiments predate the BD value model.

Figure 4.9: Interference graph of the set of 218 UHF stations within two edges of a New York station under a 126 MHz clearing target.

achieved nearly optimal efficiency at much lower cost than VCG.

4.5 Conclusions

High fidelity, computationally intensive simulations are an approach for quantitatively contrasting candidate market designs which offer a key advantage: the quality of answers provided by such simulations will improve with available computational resources. We outlined a six-step simulation methodology and instantiated it in the context of the incentive auction to investigate previously unanswerable questions about the cost and efficiency of certain alternative designs. To validate the robustness of our results, we used two quite different value models: one from the empirical economics literature and another that we constructed to rationalize public bid data. Our main findings were that: repacking VHF led to significantly lower costs and more efficient outcomes; the multiple stage clearing rule substantially both increased costs and reduced efficiency; a simple amendment to the clearing algorithm could nearly eliminate multi-round inefficiency; the performance of pops scoring relative to other scoring rules varied widely based on the value model and whether the VHF band was repacked; the specialized feasibility checker developed

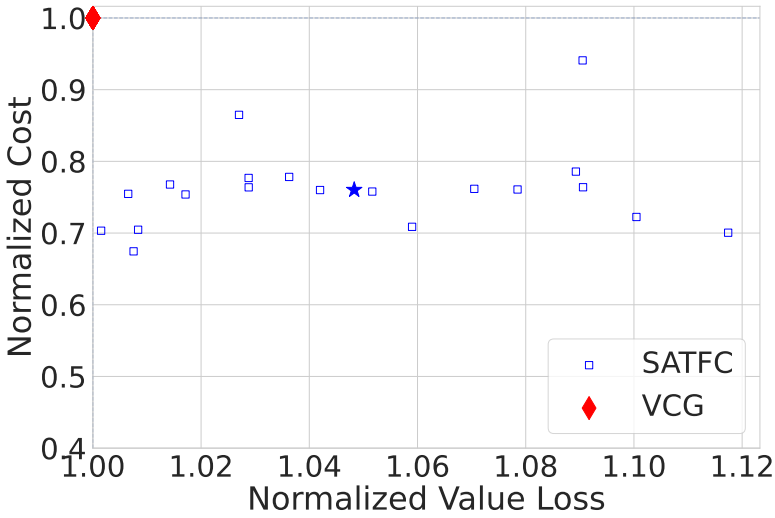


Figure 4.10: Comparing VCG auctions and reverse auctions in Greater New York City simulations.

for the auction significantly improved both cost and efficiency; alternative bid processing algorithms, while helpful, would not have made a significant difference to auction outcomes; and the reverse auction outcomes in regional simulations had efficient outcomes. We hope these specific insights can help to inform future auction designs.[20] More broadly, our analysis demonstrates the practicality[21] and promise of large-scale computational analysis of the simulated behavior of candidate market designs in highly complex settings.

[20] Kiddoo et al. [2019] remark: "Numerous government regulators around the world have sought FCC input on how they could apply market-based mechanisms such as the Incentive Auction to clear spectrum"

[21] Altogether, our final experiments took more than 60 core-years to run. Writing this chapter consumed perhaps 200–300 core-years of compute in total, since we often found ourselves needing to rerun experiments as we uncovered bugs, changed parameters, or refined our experimental questions. We do not provide these CPU time numbers with the goal of impressing the reader with how much compute power we used. Instead, we hope (a) to give other researchers a sense of the scale of experiments that we consider necessary for answering questions like the ones we tackled here, and (b) to reassure the reader that we ran more or less the largest experiments that were practically feasible.

Chapter 5

The Positronic Economist: A Computational System for Analyzing Economic Mechanisms

Mechanisms are complicated to analyze because participants respond to rule changes strategically. The previous chapter relied on a combination of obviously dominant strategies and heuristics for simulating bidder behavior. We now ask what can be done for mechanisms where the designer does not know in advance what behaviors are reasonable to simulate. Unfortunately, game theoretic analysis is a difficult process requiring either substantial human effort or very large amounts of computation. The need to solve for agent behavior limits the complexity and size of mechanisms that can be studied without making undue sacrifices in fidelity to achieve tractability (a tradeoff mostly not present in the simulations approach of the previous chapter). In this chapter, we'll restrict ourselves to one-shot simultaneous move games with payoff structures amenable to a particular compact representation. Equilibrium solvers can leverage this compact representation to tackle a larger class of games than would otherwise be possible under payoff matrix representations.

5.1 Introduction

There have been recent advances in algorithms for efficiently computing game-theoretic solution concepts [Jiang et al., 2011, Roughgarden and Papadimitriou, 2008], and on using such algorithms to analyze mechanisms [Thompson and Leyton-Brown, 2009], an approach referred to as *computational mechanism analysis* (CMA). While these algorithms can operate on normal-form descriptions of games, applying them to computationally motivated *compact game representations* allows for exponential speedups [Jiang et al., 2011, Kearns et al., 2001]. For example, Action-Graph Games (AGGs) can compute an agent's expected utility under an arbitrary mixed strategy profile with a polynomial-time dynamic programming algorithm, and this expected utility problem constitutes the inner loop of many game-theoretic algorithms. (The standard method of computing expected utility is polynomial in the size of the normal form; since an AGG can be exponentially smaller, the AGG representation can yield exponential speedups.)

This approach of combining efficient algorithms with compact representations has been used to address open problems in mechanism analysis ranging from bidding in advertising auctions [Thompson and Leyton-Brown, 2018, 2009] to strategic voting in plurality elections [Thompson et al., 2013]. The approach has the advantage that games need not be approximated and that equilibria are computed exactly; we leverage compact representations and sophisticated, exact-equilibrium-finding algorithms to manage computational demands. However, we have found it nontrivial to produce the application-specific encodings into compact game representations upon which this process depends.

This chapter's goal is to make CMA accessible to a broader audience without reducing its fidelity. We summarize our desiderata as the "Three Laws of Positronic Economics," inspired by Asimov's [1942] "Three Laws of Robotics."

1. **Precision:** Games are represented exactly. Equilibria are either exact or are ϵ-equilibria with ϵ roughly machine-ϵ.

2. **Speed:** Algorithms must be fast enough to be used in practice, except when this is in conflict with the first law.

3. **Autonomy:** Human effort should be minimized, except when this conflicts

with the first two laws.

In this chapter, we introduce a tool, *Positronic Economist* (PosEc), that delivers on these goals, dramatically reducing the human effort required by CMA without compromising the analysis itself. Our chapter makes two major contributions. The first is a high-level Python-based declarative language for describing mechanisms in a form that closely resembles their natural mathematical representations; the second is a pair of complementary algorithms that automatically infer the structure of a game specified in this language and produce a compact Bayesian AGG (BAGG). We then draw on a portfolio of existing algorithms to solve the game. We chose to use BAGGs because of the availability of empirically fast tools available for working with them: while Nash equilibrium finding is an NP- or PPAD-hard problem for BAGGs, depending on the equilibrium type, good heuristics can often find exact equilibria quickly [Jiang et al., 2011, Thompson et al., 2011].

The structure of the remainder of this chapter is as follows. Section 5.2 surveys related work. Section 5.3 formalizes the mechanisms and settings that PosEc can represent. Section 5.4 briefly describes BAGGs. Section 5.5 provides an overview of the PosEc representation language. Section 5.6 details PosEc's two structure inference algorithms. Section 5.7 discusses experiments that help characterize PosEc's performance. Finally, Section 5.8 discusses some directions for future work.

5.2 Related Work

We are not aware of any other work with the goal of simplifying the process of generating compact representations. Perhaps the closest is by Duong et al. (2009), which also integrates with the compact games literature. They provide algorithms to construct a graphical game that best approximates an input game from a set of samples of strategies and payoffs, which either come from previously observed play or a simulator. The question of how to obtain samples from a simulator without unacceptably degrading fidelity is unanswered. We note that PosEc does not require any samples.

We are aware of two other CMA systems, which differ from PosEc in the human effort required to specify games, in the types of games that they are able to

specify, and in their ability to leverage high performance algorithms. The system of Rabinovich et al. (2013) does not explicitly specify how games are represented. A user must provide code for computing expected utility given a strategy profile, which could require substantial human effort; observe that one of the main benefits of AGGs is that they provide such an efficient computational procedure. Their system is restricted to two player games of incomplete information, places restrictions on the form of utility functions, and requires strategy and type sets to be continuous and unbounded. While PosEc cannot handle infinite strategy or type sets due to the discrete nature of BAGGs, it can represent games with any number of players and does not impose restrictions on the form of the utility function. Their system only supports the fictitious play algorithm (FP), which is a relatively weak Nash equilibrium computation algorithm, prone to getting stuck in cycles.

Another CMA system was introduced by Vorobeychik et al. (2012). This system describes mechanisms and settings as piecewise linear equations. Given that many single-parameter mechanisms and settings are described algebraically in the literature, this representation requires very little human effort. However, the only supported equilibrium-finding algorithm is iterative best response (IBR), which is unable to compute mixed-strategy equilibria and can also fail to converge. In contrast, PosEc makes it easy for the user to leverage a wide range of high-performance solvers.

There is other work in the empirical game-theoretic analysis (EGTA) space that proposes specialized algorithms that operate directly on a simulator, without translation to an intermediate form. One example is by Vorobeychik and Wellman (2008), who used simulated annealing to find a strategy profile constituting an approximate Nash equilibrium of the simulated game. While the human effort of building a simulator is comparable the effort of specifying a game in PosEc, the major difference is that PosEc creates a compact game that represents the input game exactly and computes exact equilibria of the input game. While approximate equilibria are a major subject of research in algorithms and complexity, researchers in mechanism design and auction theory overwhelmingly favor exact equilibria. One difficulty with using algorithms that produce approximate equilibria is that it is often difficult to compute the ϵ for which the strategy is an ϵ-equilibria; [Bosshard et al., 2020] present an algorithm for computing ϵ-Bayes-Nash equilibria in com-

binatorial auctions that comes equipped with a verification step that upper bounds ϵ.

5.3 Mechanisms and Settings

We now formally describe the games that PosEc is able to represent. An *epistemic-type Bayesian game* is specified by $\langle N, A, \Theta, p, \mathcal{U} \rangle$, where N is a set of agents, numbered 1 to n, $A = A_1 \times \cdots \times A_n$, where A_i is a set of actions that agent i can perform, Θ is the set of private types that an agent can have, p is the joint type distribution, $p \in \Delta(\Theta^n)$ where Δ denotes the set of probability distributions over a given domain, and $\mathcal{U}$ is a profile of n utility functions where $\mathcal{U}_i : A \times \Theta^n \to \mathbb{R}$.

This chapter considers "mechanism-based games," and so splits games into two parts, a mechanism and a setting. A *mechanism* is given by $\langle A, M \rangle$ where $A = A_1 \times \cdots \times A_n$, where A_i is a set of actions that agent i can perform, and M is the choice function, $M : A \to \Delta(O)$. A *Bayesian setting* is given by $\langle N, O, \Theta, p, u \rangle$ where N is a set of agents, numbered 1 to n, O is a set of outcomes, Θ is the a set of private types that an agent can have, p is the joint type distribution, $p \in \Delta(\Theta^n)$, and u is a utility function $u : N \times \Theta^n \times O \to \mathbb{R}$. Any mechanism and setting that both use the same n and O can be combined to form a game where $\mathcal{U}_i(\theta_N, a_N) = u(i, \theta_N, M(a_N))$ and where $a_N \in A$ denotes an action profile.

5.4 Bayesian Action-Graph Games

PosEc uses Bayesian Action-Graph Games to represent the games described in Section 5.3. We briefly and informally introduce BAGGs here, but refer the reader to Jiang and Leyton-Brown (2010) for more details. BAGGs are compact because they exploit *anonymity* (an agent's payoff may not depend on the specific identities of agents who played certain actions) and *context-specific independence* (an agent's payoff for playing a given action can be determined based only on the distribution of play over a strict subset of the actions). A BAGG is a directed graph in which the nodes correspond to type-action pairs. Play of the game can be thought of as each agent placing a token on one of their allowed subset of nodes. Given the locations of all of the tokens, an agent's utility can be computed by referring only to the count of tokens in the *neighborhood* of the node the agent chose. A

node's neighborhood is the set of nodes having outgoing edges that point to it, with self-edges allowed. The counts of all of the nodes in a neighborhood are called a *projected configuration*, and each node stores a payoff table indexed by these configurations. We conclude by mentioning *function nodes*: no agent selects these nodes; instead, the count at a function node is an arbitrary deterministic function of the counts of the node's parents. For example, a sum node in a voting game might be used to count all of the agents that choose actions that influence a particular candidate's score. Since expected utility computation runtime depends asymptotically on the in-degree of action nodes, and this can be substantially reduced via the use of function nodes, they can lead to large computational savings.

5.5 Representing Games with PosEc

PosEc is a language that aims to make it easy for users to describe mechanisms and settings. The package is open source and pointers to the software, compatible equilibrium-finding algorithms, and further documentation are available at https://www.cs.ubc.ca/research/posec/. We now discuss some of the key decisions that went into its design.

Our modeling language is based on Python. The tuples, sets and utility functions of the mathematical representation map quite naturally to tuples, set and functions in Python. To specify a mechanism-based game, a user must create and combine a *setting* and a *mechanism*.

Figure 5.1 shows all the code that is required to define the mechanism and setting for plurality voting with randomized tie breaking. First a Bayesian setting is defined as consisting of ten agents, three outcomes (one in which each candidate wins), types corresponding to unique preference orderings, a uniform probability distribution for each agent over each type, and a utility function that returns the index of the elected candidate in the agent's preference ordering. Second, an action function `A` is defined as returning the actions available to a given agent (here, to vote for any candidate) and is combined with a choice function `M` that returns a distribution over outcomes given each agent's actions.[1] Finally, the `makeAGG`

[1]Note that PosEc requires randomized mechanisms, like our tie-breaking scheme, to return distributions over outcomes rather than performing randomization internally (e.g., via the Python `random` module). If choice functions randomized on their own, PosEc would need to sample the choice func-

```
# Setting
n = 10
O = ("A", "B", "C")
Theta = [("A", "B", "C"), ("C", "B", "A")]
P = [UniformDistribution(Theta)] * n

def u(i, theta, o, a_i):
  return theta[i].index(o)

s = BayesianSetting(n, O, Theta, P, u)

# Mechanism
def A(setting, i, theta_i):
  return setting.O

def M(setting, a_N):
  scores = {o: a_N.count(o)
            for o in setting.O}
  maxScore = max(scores.values())
  winners = [o for o in scores.keys()
             if scores[o] == maxScore]
  return UniformDistribution(winners)

m = Mechanism(A, M)
agg = makeAGG(s, m, symmetry=True)
```

Figure 5.1: Sample code defining a mechanism and setting for plurality voting with randomized tie breaking.

function is called,[2] and the structure inference algorithms described in Section 5.6 produce a BAGG.

One of our goals was to let users implement their utility and choice functions however they liked. Indeed, PosEc will convert any valid Python functions into a valid BAGG. However, as will be made more explicit in Section 5.6.1, PosEc can take advantage of game structure signaled through appropriate use of PosEc's *accessors*, which are used by the choice and utility functions to get information about agent types and actions played (i.e., any usage of `a_N` or `theta`). Note the call to `a_N.count(o)` in the choice function, which signals that the outcome can be determined based only on the *number* of agents that played a given action. Consider an anonymous mechanism with a constant number of actions c, like our

tion and would then only approximate the distribution.

[2] `symmetry=True` specifies that agents of the same type share the same action node set.

voting game. The corresponding normal-form game requires $O(nc^n)$ space, while the corresponding AGG is $O(n^c)$.

In many single-good auction settings, each agent's payoff depends only on whether she is allocated the good and on her own payment. Such structure is computationally useful: an agent's utility may be computed without deriving a distribution over the entire outcome space. We call this idea *projection*. PosEc allows users to specify projected settings and mechanisms. While expressing projection structure can be more work for users, doing so can yield exponentially faster computation, because the resulting games can be much more compact than games based on the equivalent (unprojected) settings and mechanisms.

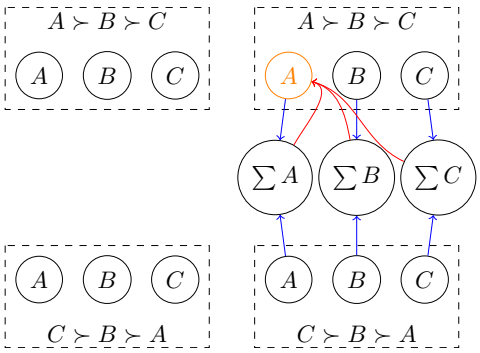
The overriding goal for the PosEc API is to allow users to precisely specify games as easily as possible. Thus, our design decisions emphasize a simple general language for users to build with, rather than a palette of options for users to choose from. Naturally, we hope that users will produce a library of reusable mechanisms and settings.

5.6 Structure Inference Algorithms

The second main component of PosEc is structure inference: automatically generating compact BAGGs given setting and mechanism descriptions in PosEc's own modeling language. We provide two approaches for doing this. First, *white-box structure inference* (WBSI) uses structure made explicit in the PosEc representation—e.g., via use of the count operator, projection, etc.—to generate a BAGG. (In the degenerate case, no such structure is explicitly given, and we obtain an exponential-size BAGG.) Second, *black-box structure inference* (BBSI) takes the BAGG generated in the first step and probes it to find additional structure to obtain a more compact BAGG.

5.6.1 White-Box Structure Inference

We aim to obtain what we call the *straightforward BAGG*: a BAGG that contains only those function nodes and edges that are necessary to compute features used by the input game. Let ℓ_s denote the representation length of this game. Our goal is to compute the straightforward BAGG using only poly(ℓ_s) calls to the utility function



(a) **(b)**

$\sum A$	$\sum B$	$\sum C$	Expected Payoff
1	0	1	1
1	1	0	1.5
2	0	0	2

(c)

Figure 5.2: WBSI's first steps on the plurality voting example in Figure 5.1. Beginning with a disconnected action graph (a), WBSI selects an action node (here, voting for candidate A with type $A \succ B \succ C$) and creates an empty payoff table. As the choice function is run, WBSI will infer from accessor calls such as `a_N.count("A")` that it needs to create summation nodes aggregating the number of players that voted for each candidate across types, and that these function nodes should be inputs to the payoff table. (b) shows the action graph once the payoff table is computed, and (c) shows the payoff table (for a 2 agent game).

of the input game. We do so via a relatively direct algorithm. Essentially, it works by beginning with a totally disconnected action graph, and progressively adding function nodes and edges whenever their absence means that the utility function cannot compute a payoff. We give pseudocode for WBSI in Algorithm 1 and work through the first steps of WBSI on the plurality voting example from Section 5.5 in Figure 5.2.

We now provide two theorems about the behavior of WBSI: the first relates the size of the BAGG produced by WBSI to parameters of the game under certain conditions, while the second relates the runtime of WBSI to the size of the

straightforward BAGG.

Theorem 1 *For any game parameterized by the number of agents n, the number of actions per agent m, and the number of types t, where the choice and utility functions each can make a constant number of different calls to PosEc accessor functions, and where any weighted-sum calls involve a maximum weight of w bounded by poly($n \cdot m \cdot t$), the straightforward BAGG will require only poly($n \cdot m \cdot t$) space.*

Proof sketch. WBSI introduces at most one function node per accessor call. A weighted max node, which returns an action node index, can only have as many different projected configurations as there are action nodes, of which there are at most $O(n \cdot m \cdot t)$. A weighted sum node with maximum w can only have at most $O(w \cdot n)$ different projected configurations. For each action node, there are a constant number of neighbors. Thus the possible projected configurations on the neighborhood of any action node is at most the Cartesian product of the possible projected configurations of each neighbor. Since these spaces are all poly$(n \cdot m \cdot t)$ and there are boundedly many of them, the total configuration space in the neighborhood of every action is at most poly$(n \cdot m \cdot t)$. □

Small outputs are important because the BAGGs produced by WBSI are typically used as inputs to game-solving algorithms, which often require worst-case time exponential in the size of their inputs. It is also important that WBSI be fast, so as not to become the main bottleneck.

Theorem 2 *The white-box structure-inference algorithm (Algorithm 1) runs in $O(c(\ell_s)^2)$ time, where ℓ_s denotes the size of its output, the straightforward BAGG, and c denotes the amount of time that the input code requires to compute a single agent's payoff for a single type-action-profile.*

Proof sketch. The runtime is dominated by the computation of payoff tables. The outer *for* and *repeat* loops jointly take $O(\ell_s)$ time: the *for* loop runs once per action node, and each iteration of the *repeat* loop after the first one involves creating a new edge, and both actions and edges take up space in the BAGG representation. The inner *for* loop iterates over projected configurations, where one payoff per projected configuration is also part of the BAGG representation. Because this

loop only deals with BAGGs that contain weakly fewer edges than the straightforward BAGG, it can only iterate over projected-configuration spaces that are weakly smaller than the projected configuration space of the straightforward AGG. Thus, this inner loop also takes $O(\ell_s)$ time. □

As long as the outcome and utility functions passed to WBSI can be evaluated in polynomial time and make bounded numbers of calls to accessor functions, WBSI will run in polynomial time and output a polynomial-sized BAGG.

Algorithm 1 White-Box Structure Inference

```
1:  Input: Bayesian game, utility represented as a function
2:  Output: Bayesian action-graph game

3:  Create action nodes but no edges or function nodes
4:  for Each action-node a do
5:      finished ← False
6:      while not finished do
7:          Create an empty payoff table for action node a
8:          finished ← True
9:          for Each projected config c on a's neighbors do
10:             Try to compute payoff given c
11:             if Success then
12:                 Add c and payoff to table
13:             else
14:                 v ← missing accessor in computation
15:                 if No function node computes v then
16:                     Add function node and edges to compute v
17:                 Add edge from function node to a
18:                 Finished ← False
19:                 Break
```

5.6.2 Black-Box Structure Inference

The goal of black-box structure inference is to take a BAGG obtained from white-box structure inference—in the degenerate case, a completely unstructured BAGG—and return a new BAGG that more efficiently represents the same game. Thus, BBSI is a constrained optimization problem where the feasible region is the set of all BAGGs that are equivalent to the input game and the objective is to find

the smallest BAGG, measured by input length, in the feasible region. A simple, polynomial-time algorithm is to try cutting an incoming edge to an action node if this results in a strategically equivalent payoff table (e.g., in a GFP auction, an agent's payoff is unaffected by each bid less than her own). When successful, we reduce by one the dimension of that action node's payoff table. Our algorithm iterates over nodes and cuts edges as long as it is possible to do so. Pseudocode is given in Algorithm 2.

Algorithm 2 Black-box structure inference

```
1: Input: Bayesian action-graph game G
2: Output: Bayesian action-graph game G′ (created in place)

3: for Each action-node a do
4:     while There are no cut-able edges to a do
5:         Randomly select an edge e that ends at a
6:         if e can be cut without breaking strategic equivalence then
7:             Cut edge e
8:             Remove e's column from a's payoff table
```

5.7 Experiments and Results

We now present experimental evidence that our two structure inference algorithms produce compact games in a practical amount of time and that these games can be used to compute sample Nash equilibria efficiently. To evaluate our algorithms, we turned to games for which previous work has manually identified compact encodings; they happen to be perfect-information games (thus, AGGs rather than BAGGs). Specifically, we recreated GFP and wGSP position auctions games from Thompson and Leyton-Brown (2009) and two-approval voting games from Thompson et al. (2013). We generated straightforward specifications of these settings in PosEc and then ran WBSI and BBSI. For every setting, for every number of agents, we generated 10 different games. The results are summarized in Figure 5.3. For our position auction games we used 4 positions, 20 bid increments, and the Varian (2007) preference model. We varied the per-agent click-through rates, valuations, and quality scores across games. We restricted to "conservative"

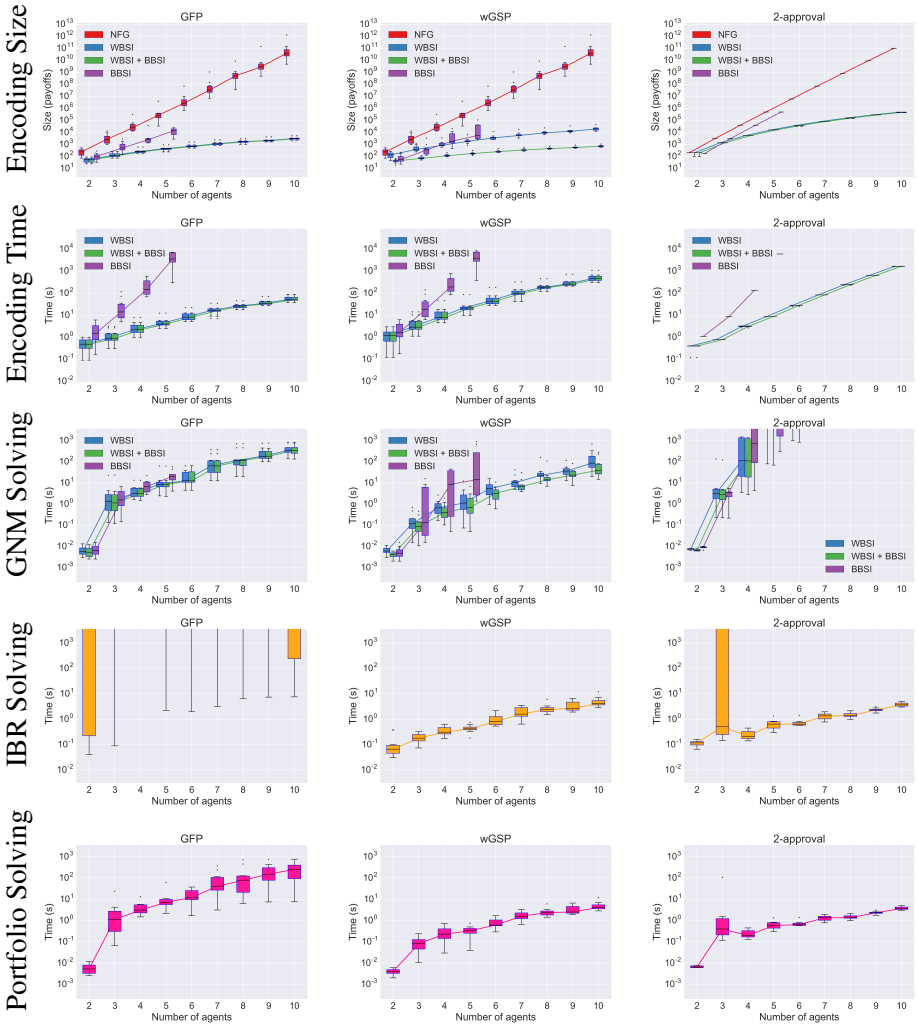


Figure 5.3: Encoding sizes (first row); Encoding times (second row); Median time required to identify an equilibrium using the GNM algorithm (third row), IBR algorithm (fourth row), and a parallel portfolio of GNM and IBR running on two cores (fifth row). WBSI+BBSI achieved substantially more compression than WBSI only in the case of wGSP; this suggests that previous work identified very effective encodings. We did not run BBSI on games with more than 5 players as the runtime required grew prohibitively large. All equilibrium finding plots are truncated at our budget of one hour; following the *penalized average runtime* (PAR10) metric, timeouts are logged as ten hours.

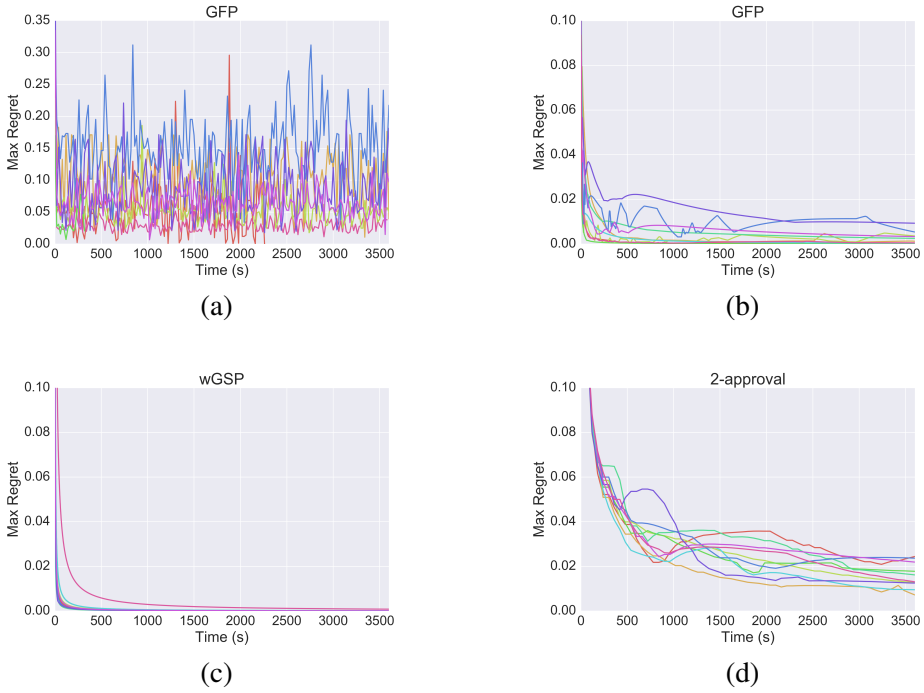


Figure 5.4: Median regret (across random initializations) achieved by IBR in (a) and FP in (b)–(c) on 10 agent games as a function of time; each line corresponds to a different game and regret is normalized by the maximum payoff in each game.

strategies [Caragiannis et al., 2011, Roughgarden and Tardos, 2012] in which bidders do not play the weakly dominated strategies of bidding above their valuations. For our two-approval games, we considered settings with 5 candidates and a variable number of voters. For each game, we randomly assigned each bidder some permutation of 0, 1, 2, 3 and 4 utility points for each of the different candidates. All of our experiments were run on Intel Xeon E5-2640 v2 processors on nodes with 96 GB of RAM.

For each game, we considered four encodings: the normal form (NFG); white box structure inference (WBSI); WBSI followed by black-box structure inference (BBSI) refinement (WBSI+BBSI); and BBSI based on the raw normal-form games. Across our three settings, WBSI always produced a BAGG which was dramatically smaller than the corresponding normal-form games. WBSI+BBSI was only able to

achieve much additional compression in the case of wGSP. BBSI was considerably worse; indeed, we were unable to test it on games with more than 5 agents because runtimes for encoding grew too large.

The runtime required for WBSI encoding was generally quite manageable, with WBSI and WBSI+BBSI growing subexponentially for GFP and GSP and exponentially for 2-approval. (Even in the latter case, runtimes only reached roughly half an hour for a setting with 10 agents and 5 candidates.) The same was not true for BBSI; runtimes grew sharply, and indeed we were unable to encode games involving more than 5 agents within a reasonable amount of time for any of our settings.

We then considered the amount of time required to identify a sample Nash equilibrium of the resulting games within a CPU budget of one hour. Recall that one of PosEc's strengths is that it is not tied to any single algorithm. We first consider Iterative Best Response (IBR) and the Global Newton Method (GNM) of Govindan and Wilson (2003), extended to BAGGs by Jiang et al. (2011). Each algorithm's runtime depends on its starting point, which we initialized randomly; we considered the median time to find an equilibrium over 10 such randomly sampled points, reporting the distribution over games for each setting. For GNM, we observed subexponential scaling for GFP and wGSP, with median runtimes of only several minutes even in the case of 10 agents.[3] In the case of 2-approval voting, we observed exponential scaling, and GNM's median runtime exceeded our CPU budget for all games with more than 6 agents.

We implemented a version of IBR that operates on the raw game specification—i.e., that does not leverage either BBSI or WBSI. Players are initialized to random pure strategies; at each iteration, a random player best responds to the current strategy (choosing among best responses at random if more than one exists). We measure the maximum regret at the end of each iteration; a regret of zero corresponds to a Nash equilibrium. IBR can only discover pure Nash equilibria. We observed that IBR timed out on most of our GFP games (see Figure 5.4, which illustrates its cycling behavior), but was quick to find an equilibrium in all of our wGSP games and in nearly all of our 2-approval games.

PosEc makes it easy to use multiple algorithms. Given the different strengths

[3] One exception: a single 4-agent GFP game was not solved for any of our GNM starting points.

and weaknesses of IBR and GNM given WBSI+BBSI AGGs, we thus considered an algorithm portfolio consisting of both algorithms running in parallel on two cores. This portfolio could find an equilibrium within our time budget for all but one four-agent GFP game.[4]

Finally, because it was used by the CMA system of Rabinovich et al. [2013] we surveyed in Section 5.2, we also consider the Fictitious Play (FP) algorithm. FP relies on expected utility calculations, and hence only runs efficiently on large games by leveraging AGG structure and the polytime expected utility algorithm of Jiang and Leyton-Brown (2006); we thus produced such an implementation based on WBSI+BBSI. We ran it on each of our 10 agent games given 10 random initializations, again tracking regret after each iteration. The results are shown in Figure 5.4. FP converged relatively quickly (though slower than IBR) for the wGSP games, but cycled or stagnated on most GFP games and all 2-approval games.

5.8 Conclusions and Future Work

This chapter makes two major technical contributions: the PosEc declarative language for describing mechanism-based games, and two structure-inference algorithms that make it possible to compactly represent such games as BAGGs. These contributions dramatically reduce the human effort necessary to perform computational mechanism analysis without leading to a substantial loss of accuracy or speed. There are many potential applications in which PosEc could shed light on hard-to-analyze economic settings. Even within the limited and well-studied sphere of single-good auctions, PosEc could be used to study non-linear utility for money (e.g., budgets; risk attitudes), asymmetric valuation distributions, other-regarding preferences (both altruism and spite), and conditional type dependence (including common and affiliated values).

One limitation of the current PosEc system is that it can only describe one-shot simultaneous-move games. In contrast, many real-world mechanisms proceed in multiple stages (e.g., sequential auctions; clock-proxy auctions). In such games, decisions made in one stage can affect which outcomes are possible or desirable

[4]Other settings might benefit from different algorithm portfolios. In particular, we note that Simplicial Subdivision [Jiang et al., 2011, van der Laan et al., 1987] and the Support Enumeration Method [Porter et al., 2008, Thompson et al., 2011] have been extended to handle BAGGs efficiently.

in the next stage. BAGGs inherently represent single-stage games, but could be used to analyze multi-stage mechanisms by representing the different stages as individual BAGGs and solving the complete system by a process of backward induction. Such a process would likely resemble the special-purpose algorithm that Paes Leme et al. (2011) proposed for computing the equilibria of sequences of single-good auctions. Alternatively, analysis could be performed using a compact game representation that explicitly supports multi-stage games, such as temporal AGGs [Jiang et al., 2009] or MAIDs [Koller and Milch, 2003]. Unfortunately, the algorithms for reasoning about such representations currently offer much poorer performance than algorithms for reasoning about BAGGs.

Another limitation of the PosEc system is the cost of explicitly representing types and actions: no BAGG can be asymptotically smaller than its number of types or actions. Thus, games with large type or action spaces—such as combinatorial auctions—cannot be succinctly represented. There is very little work on representing games with implicitly specified action spaces—Koller and Milch (2003) and Ryan et al. (2010) are the two exceptions, unfortunately both without good implementations—but as this literature develops there may be opportunities for extending our encode-and-solve CMA approach to new game families.

A third limitation is the requirement to discretize actions and (in the case of Bayesian games), types. We note that many real-world mechanisms do discretize actions; thus, we argue that PosEc's ability to give insight into such settings is also one of its benefits.

Finally, as mentioned earlier, this chapter considered only non-Bayesian-game settings, even though PosEc supports BAGGs as well as AGGs. This was simply because we wanted to compare to manual AGG encodings, which existed only in the case of perfect-information games. Future work may explore the computational cost of computational mechanism analysis in Bayesian game settings.

Chapter 6

Understanding Iterative Combinatorial Auction Designs via Multi-Agent Reinforcement Learning

As mentioned in the conclusions of the preceding chapter, a major limitation of the Positronic Economist is that it cannot be used to analyze extensive-form games. While Chapter 4 provides an example of computational mechanism analysis on a dynamic game, the analysis relies on dominant and heuristic strategies. How can one analyze, for example, the incentive auction's forward auction, a more traditional iterative, combinatorial spectrum auction? In this chapter, we investigate how to perform comparative statics in dynamic games without dominant strategies, at a scale beyond the capabilities of pen-and-paper and traditional equilibrium solvers. Similar to prior chapters, we evaluate the impact of candidate policy changes in auctions computationally, simulating the auction under different rules and then measuring differences in outcomes of variables of interest such as welfare and revenue. The key difference is that in this chapter, we discover equilibrium behavior using multi-agent reinforcement learning algorithms. We show preliminary evidence that reinforcement learning algorithms represent a viable direction

for extending computational mechanism analysis to extensive-form games.

6.1 Introduction

Iterative combinatorial auctions are a high-stakes, socially important class of mechanisms. These auctions have become the dominant mechanism for radio spectrum privatization [Teytelboym et al., 2021], though they have also been deployed in a wide range of other settings [see, e.g., Palacios-Huerta et al., 2021].

It is important for bidders to make good strategic choices in an auction whose outcome can materially affect their company's value. It is also important for designers to establish auction rules that achieve good outcomes in terms of revenue, welfare, and other metrics. However, neither task is easy, because iterative combinatorial auctions are profoundly complex. Combinatorial designs explode bidders' action spaces, but are essential when bidders have strong needs to express complementarities and/or substitutabilities across individual goods. Iterative designs yield a second exponential increase in strategic complexity, but are critical when bidders need to perform "price discovery" by updating their stated preferences in response to information about others' demand.

Consider the case of spectrum auctions. While there are a few popular auction formats, such as Simultaneous Multiple Round Auctions (SMRA) [Milgrom, 2004], Combinatorial Clock Auctions (CCA) [Ausubel et al., 2006], and Clock Auctions[1] [Ausubel, 2004], even within a single format, rules tend to differ from one auction to the next. For example, between the (consecutive) Canadian 700 MHz [ISED, 2012] and 600 MHz [ISED, 2018] auctions, the activity rule was altered; later, between the Canadian 3500 MHz [ISED, 2021] and 3800 MHz [ISED, 2022] auctions, a spectrum cap was added limiting the maximum a bidder could win in each region. Given that rules are in constant flux, and that it is difficult to predict how isolated rule changes will impact an auction's equilibrium, it is important to develop methods for reasoning about the effects of counterfactual policy changes in these auctions.

Ideally, we would analyze such problems by computing the Bayes–Nash equi-

[1] Clock Auctions are combinatorial auctions, but are a distinct auction format from Combinatorial Clock Auctions.

libria of the game induced by each set of auction rules, evaluating how outcomes change in equilibrium. This is the dominant approach taken by theoretical work in economics, and has also been adopted by various computational approaches [e.g. Rabinovich et al., 2013, Thompson et al., 2017, Thompson and Leyton-Brown, 2017]. However, there is little hope of identifying Bayes–Nash equilibria of realistic iterative combinatorial auctions, as they are too complex to admit pen-and-paper analysis using known techniques. Thus, the most influential existing work studies highly simplified settings. For instance, for one auction format, Riedel and Wolfstetter [2006] proved that it is an equilibrium for bidders to end the auction in the first round, but their proof applies only to auctions with a single product, infinitesimal price increases, and complete information about bidders' values. Combinatorial action spaces and multi-round structures imply enormous extensive-form representations, putting them out of reach of traditional equilibrium solvers as well.

An alternative (adopted in Chapter 4) is to simulate the auction under the assumption that bidders play some given, fixed strategy [e.g., Cary et al., 2007, Newman et al., 2024]. One can then observe both the effect of a rule change and its sensitivity to changes in bidders' valuations and to randomness in the auction mechanism. However, combinatorial auctions are not typically strategyproof [see e.g., Bosshard and Seuken, 2021, Levin and Skrzypacz, 2016]. There is thus typically not a singular reasonable strategy to simulate, let alone one with consistent incentive properties across the rules being studied.

Multi-agent reinforcement learning (MARL) offers a middle ground: agents who exhibit much richer strategic reasoning than in static simulations but much more computational tractability than classical Bayes–Nash equilibrium computation. Over the past decade, MARL algorithms have made great strides. Early efforts focused on achieving superhuman performance in two-player zero-sum games, such as Go [Silver et al., 2016], Shogi [Silver et al., 2017], and Stratego [Pérolat et al., 2022]. These games share the special property that players do not need to solve the equilibrium selection problem [Harsanyi and Selten, 1988]: if a player plays their part of an equilibrium strategy, it will do at least as well against any other strategy an opponent might play. Of course, this property does not hold in auctions, nor does it hold in zero-sum games having more than two players. It is thus striking that recent work has shown superhuman performance in multi-

player zero-sum games such as poker [Brown and Sandholm, 2019a]. Of particular relevance to what follows, Counterfactual Regret Minimization (CFR) [Zinkevich et al., 2007] is a popular, modern MARL algorithm that was recently used to solve Texas Hold 'em poker [Bowling et al., 2015, Brown and Sandholm, 2018]. Another impressive recent achievement was the demonstration of human-level play in the board game Diplomacy [Anthony et al., 2020, Bakhtin et al., 2022]. Indeed, this game shares other key similarities with our domain: a combinatorial action space and iterative, simultaneous moves.

We foresee no near-term interest from market participants in deploying "superhuman" bots to bid autonomously in complex and high-stakes auctions without human oversight—nobody is going to bet a Fortune 500 company's future on an RL algorithm running against unknown opponent strategies in a novel economic environment. However, even in such domains, MARL tools offer promise for evaluating competing auction designs and gaining economic understanding. In this vein, MARL has been used to identify and evaluate complex policy decisions regarding taxation [Zheng et al., 2022] and regulations on economic platforms [Wang et al., 2023]. MARL has also been applied to bargaining, where prior work has studied how different learning algorithms lead to more or less socially desirable outcomes [Abramowicz, 2020, Li et al., 2023]. We are aware of fewer papers that use RL to study auctions specifically. Banchio and Skrzypacz [2022] study the behaviour of Q-learners in first- and second-price auctions, arguing that in fast-paced online advertising auctions such algorithms really do interact with each other and hence it is more important to understand their joint dynamics than to consider equilibrium behavior. Deep MARL has also been used to compute equilibria of single-round combinatorial auctions [Bichler et al., 2021, 2023] and multi-round combinatorial auctions with continuous valuation and action spaces [Pieroth et al., 2023, Thoma et al., 2023a,b]. Closest to the domain we study, Pacaud et al. [2023] combine heuristic strategies with Monte-Carlo tree search to find effective bidding strategies in clock auctions; however, they focus on finding strategies that perform well against existing heuristics in auctions with complete information.

This chapter aims to use MARL to gain economic insight in (incomplete-information) iterative combinatorial auctions. Such insights could help a human bidder assemble a strategic "playbook" by providing examples of strong bidding

both by an agent with their own preferences and by their counterparties; to date, such playbooks are typically derived via pen-and-paper analysis of dramatically simplified subproblems and via observations of expert human play in a handful of "mock auctions". Insights gleaned from MARL analysis could also help an auction designer to trade off the costs and benefits of candidate rule changes, evaluating economic variables such as revenue, welfare, length of the auction, variance in outcomes, etc. Unfortunately, actually running MARL to understand an iterative combinatorial auction is not nearly as simple as implementing an auction simulator and then unleashing multiple copies of some off-the-shelf algorithm. Modeling a spectrum auction verbatim from its rulebook produces games that are vastly too large to solve. One is then immediately faced with important modeling choices, finding ways to reduce the game to a feasible size without abstracting away the most strategically important elements.

Our work makes two main contributions. First, in Section 6.2, we present a general methodology for studying complex, iterative auctions using MARL. Notably, we advocate for simplifying the auction model by abstracting the action space, controlling the number of bidders, and limiting the auction's length, without giving up on qualitatively important features such as imperfect information or asymmetry between bidders. We describe the landscape of MARL algorithms and important problems that can arise, such as arbitrary behavior when multiple actions lead to the same outcome, and convergence to brittle equilibria that involve unrealistic levels of coordination. We also discuss key challenges in verifying an algorithm's convergence, and interpreting trained policies in the face of multiple equilibria.

Second, we present a case study showing that, despite the challenges just described, MARL can be used to conduct novel, economically meaningful analysis. We consider a prominent family of iterative combinatorial auctions called clock auctions and, leveraging two modern MARL algorithms, investigate a key design choice: how to process bids when multiple bidders want to drop the same product to a level below supply. Section 6.3 describes the problem, how we model it as a game, how we solved it using MARL, and how we assessed performance. Section 6.4 then goes on to demonstrate that this rule change does indeed give bidders incentive to change their behavior, yielding substantial differences in both revenue

and auction length. Furthermore, we show that modeling bidders as following a straightforward, myopic bidding heuristic gives misleading results in the same setting, and indeed would lead a designer to conclude that the rule change has the opposite effect.

Beyond these findings, we contribute a highly configurable clock auction environment. In the hopes that it will serve as a testbed for future research on MARL for auctions, we expose many parameters representing possible auction rule changes. We also provide software that generates clock-auction game instances and associated bidder values, leveraging a realistic value model from the literature [Weiss et al., 2017]. All of our code is available at https://github.com/newmanne/open_spiel. The chapter concludes in Section 6.5, where we discuss various design choices ripe for future investigation and speculate on methodological advancements.

6.2 Methodology: Understanding Iterative Combinatorial Auctions via Multi-Agent Reinforcement Learning

We begin by laying out our methodology for applying MARL to understand iterative, combinatorial auctions. First, we discuss how we model the auction, trading off model fidelity with size of the resulting game tree. We then turn to "solving" the game using MARL, discussing the strengths and weaknesses of different MARL algorithm families, how to address the problem of multiple equilibria, and considering equilibrium refinements (preferring pure equilibria; avoiding brittle equilibria). The section concludes by considering how to validate and interpret policies: assessing the extent to which policies have converged to a Bayes–Nash equilibrium and reasoning across multiple equilibria.

6.2.1 Modelling an Auction

It is tempting to simply model an auction wholesale, mapping its entire rules manual into a MARL environment, and then unleashing a MARL algorithm. Unfortunately, this approach is very unlikely to succeed for iterative combinatorial auction problems because the number of bundles will be large, the number of bidders will

be large, and the auction will last many rounds, yielding an enormous game tree that is simply infeasible to solve. Much like pen-and-paper analysis, one must abstract away certain elements to buy the ability to accurately model others.

Certain kinds of rule complexity that would pose significant barriers to pen-and-paper analysis are unproblematic for MARL, such as winner determination algorithms that fall back on default behaviors when solvers time out, ideosyncratic tie-breaking rules, and complex activity rules determining conditions under which certain actions are available to certain bidders. In other cases, auction elements can dramatically increase the number of actions available to a player in each information set, such as opportunities for intra-round bidding, the option to sidestep activity rules a small number of times, or the ability to place jump bids.

We recommend gaining a solid foundation by starting with simple, easy-to-solve games and scaling up later rather than starting with complex ones that might prove to be intractable. We have been surprised countless times by how even auctions with moderate numbers of bidders, goods, and rounds can lead to an enormous, unwieldy game tree, leaving little hope of obtaining reliable results from MARL algorithms. However, as we demonstrate in our experiments in Section 6.3, there is a sweet spot of game sizes out of reach of pen-and-paper analyses, but very much tractable for MARL. We begin by describing several high-level decisions that a designer can make to control the size of the game tree. The key metric of importance is usually the number of distinct *information states* in the game—the number of game states that an agent can distinguish.

Number of actions.

Combinatorial auctions typically have enormous action spaces, often corresponding to the set of all legal bundles. The action space determines the branching factor of the game tree, so reining it in has a dramatic effect on game size. One helpful way to limit the size of an action space is to experiment with auctions that have few goods or units for each good available, limiting the number of possible bids. However, we caution against reducing the auction down to a single product (e.g., several units of a generic license), as auctions with only one product may not engage with many of the auction's rules, leaving interesting behavior unstudied.

It is also possible to study auctions with more available goods by restricting bidders more heavily, only allowing them to choose from a small number of heuristic strategies at each state. For example, these heuristics could include bidding myopically, maintaining the last round's bid, raising prices on an opponent's holdings, or attempting to drop out of the auction; of course, many other options are possible. Care is required here to ensure that this restriction does not significantly hamper players' ability to best respond to others' strategies. As a concrete example, one might reduce the action space by limiting bidders to only bidding on packages they genuinely value. However, it is well-known that there are auction instances where such *simple bidding* is never optimal in Bayes–Nash equilibrium [Bosshard and Seuken, 2021]. A helpful intermediate option is to allow players to deviate from a given set of heuristics up to k times per auction; this can be useful when it is only strategically important to deviate from known heuristics occasionally, and also can be used to diagnose the effectiveness of the given set of heuristics. This approach has the beneficial side effect of producing strategies that are easier for an analyst to interpret, which can be particularly helpful for developing a playbook. Restricting actions to a smaller set is also likely to make the resulting strategies more useful as meta-strategies for use in Empirical Game Theory Analysis [Wellman, 2006].

In some auctions, a naive implementation would produce a continuous action space. For instance, both intra-round bidding in clock auctions and the supplementary rounds of CCAs give bidders the ability to report real-valued prices. While there is some work on RL with continuous actions, the simplest method, which we adopt in this work, is to simply avoid modelling the part of the auction that admits a continuous action space. When this is not possible, a good alternative is to restrict possible bids to a fixed grid: for instance, allowing bidders to only bid in \$1 increments. We suggest experimenting with multiple discretizations that vary in granularity to ensure that the chosen discretization does not have a substantial effect on the results. Finally, one can again employ a finite set of heuristics (each of which maps to an unrestricted real value), recovering a discrete action space without discretizing bids themselves.

Auction length.

Real-world spectrum auctions can last tens to hundreds of rounds. This is problematic for MARL analysis, as the size of a game tree typically grows exponentially with its maximum length: with r rounds and B possible bids each round, a game tree would have B^r states! However, there is often little value in analyzing such long auctions. In practice, it is common to see the same bids repeated at slightly increasing prices for many rounds, with much of the action compressed into a few key rounds. We recommend setting opening prices and price increments such that auctions last no longer than ten rounds; this tends to avoid enormous game trees without sacrificing much of the strategic complexity of the auction. In some cases, even two or three rounds can be enough to yield economically meaningful conclusions.

A related technical issue is that, in theory, auctions can be arbitrarily long: as long as bidders continue to overdemand products, the auction will continue. These arbitrarily large game trees do not typically pose a problem to theorists, who can handle them with inductive proofs, nor to reasonable bidders in practice, who would never expose themselves to indefinitely rising prices. However, they *do* pose a problem to most RL algorithms, which cannot learn to avoid each of these unreasonable bid trajectories without first gaining experience by making them. We recommend constraining agents' bids to rule out such arbitrary-length bidding sequences altogether. A natural solution is to disallow bidders from making bids that are strictly dominated by dropping out of the auction, ensuring that they would eventually be forced to drop out when prices become sufficiently high. Another is modelling bidders as budget-constrained, disallowing bids that would exceed a bidder's budget if processed. Budgets are a more tenuous modelling choice: while real-world bidders do truly have finite resources, it may be unrealistic to prohibit a bidder from buying a license that they value more than its cost. A further concern is that making an over-budget bid does not mean that the bidder will necessarily end up paying more than their budget; preventing such moves may unwittingly restrict the set of equilibria. We also experimented with implementing a fixed limit on auction length, giving all players arbitrary, negative utilities upon hitting this limit; we recommend against this as it can have a dramatic effect on the equilibria

of the game.[2]

Number of bidders.

In practice, spectrum auctions can involve dozens of bidders. However, many bidders in real auctions represent small, locally-operating telecoms that have limited ability to make large-scale bids, and relatively few bidders make strategic choices at a national scale. If smaller bidders need to be modeled, consider whether they need the flexibility of strategic adaptation, whether their action spaces can be restricted to their bundles of interest, or whether a deterministic heuristic strategy would suffice.

Perfect or imperfect information.

In a perfect-information game, each bidder knows their opponents' exact valuations. Perfect-information models of auctions often have equilibria in which bidders coordinate strongly and unrealistically. For example, in the perfect-information model of a sealed-bid, first-price auction, the bidder with the highest value bids the second-highest bidder's value and all other bidders bid arbitrarily; missing much of the strategic reasoning that takes place in such an auction. Analogously, when we experimented with perfect-information models of iterative combinatorial auctions, we often found that they terminated in a single round.

Real auctions, where bidders likely have priors over their opponents' strategic goals and valuations, are best modelled as Bayesian games. While Bayesian modeling is a big step mathematically, in an RL environment it amounts simply to adding a chance node at the root of the game tree. Such a node creates a copy of the game tree for each type combination, which leads to a much larger game tree but, since bidders do not know each others' types, has a smaller effect on the number of information states. We thus recommend experimenting with bidders that have a small number of types. For example, when we adapted the experiments just described to imperfect-information settings in which each bidder can have one of multiple possible valuations, we have typically observed more interesting, multi-

[2]Specifically, we found that weak bidders that would otherwise be allocated nothing could make non-credible threats to bid up to the limit, forcing stronger bidders to concede an item.

round behavior, including pooling equilibria in which weaker types act like their stronger counterparts to drive better negotiations than they "deserve".

Asymmetry between bidders.

It is common in auction analysis to model bidders as symmetric. This offers several technical advantages. When all bidders are symmetric, the equilibrium selection problem can mitigated by searching for symmetric equilibria. Furthermore, if all agents behave symmetrically, only a single policy needs to be trained. However, in reality, bidders in spectrum auctions are not usually well-approximated as symmetric, as they often have vastly different resources, business interests, and priorities. Thankfully, such asymmetry is a straightforward addition to a MARL environment, as RL algorithms have little trouble dealing with asymmetric bidders. We advocate for experimenting with varying bidders' relative strengths.

6.2.2 Finding Equilibria

We now turn to the problem of using MARL to find near-equilibrium strategy profiles.

Choosing an algorithm.

At a high level, MARL algorithms vary along two key dimensions. The first is whether they use a lookup table or function approximation to represent players' policies. In tabular methods, each information state in the game is represented separately; bidders faced with a new information state do not reason about its similarity to other states with which they have previous experience, but instead start learning afresh. This is a double-edged sword: it does not constrain players' behavior in any particular way, allowing them to learn flexible strategies, but causes erratic behavior (typically close to uniform random) in very rarely encountered states. With function approximation (i.e., deep reinforcement learning), each information state is represented by a feature vector and the agent's behavior is regularized to be similar in states having similar feature vectors. This allows agents to generalize from one part of the game tree to another, but can be more difficult to train reliably. Some tabular methods, such as Counterfactual Regret Minimization [Zinke-

vich et al., 2007], also assume *perfect recall*—that an agent never forgets anything that they previously observed—while function approximation methods generally do not. Consider whether it seems strategically important that agents never blend observations together.

The second key dimension distinguishing MARL algorithms is which parts of the game tree are explored in each training iteration. Many algorithms from single-agent reinforcement learning, such as Q-learning [Watkins and Dayan, 1992] or policy-gradient methods [Sutton et al., 1999], only sample a single path through the game tree at each iteration. While it is possible to use these algorithms in multi-agent settings by running them independently for each player, they can struggle due to focusing most of their training effort on a small fraction of the game tree. At the other extreme, algorithms such as Counterfactual Regret Minimization (CFR) [Zinkevich et al., 2007] enumerate every information state in each iteration, getting good coverage over the entire game tree, but their runtime quickly becomes infeasible for larger games. Lastly, some algorithms fit between these two extremes, enumerating one player's actions but sampling opponent actions and chance outcomes; one such example is external-sampling Monte-Carlo CFR (MCCFR) [Lanctot et al., 2009].

If it is feasible to enumerate the entire game tree, we see little reason not to use a tabular method that explores much of the game tree, such as CFR or related variants; at the very least, we recommend such algorithms as a baseline for comparing to other approaches. For very large games, where it becomes infeasible to run even a single iteration of a CFR variant, function approximation methods are likely the way forward.

Multiple equilibria.

Complex auctions typically admit multiple equilibria. When using non-deterministic MARL algorithms, a simple way to find multiple equilibria is to run the algorithm with multiple random seeds, which influence both its random initialization and any random events within the RL environment, causing it to take a different learning trajectory on each run. We thus recommend non-deterministic algorithms, e.g., preferring Monte-Carlo sampling variants of CFR over the original algorithm. We

have less experience with deterministic MARL algorithms; we speculate about potential approaches to identifying multiple equilibria with such algorithms in Section 6.5.2.

When two or more actions produce the same terminal reward, RL algorithms have no reason to prefer one action over another, and equilibria multiply. This can be detrimental to both convergence and auction outcomes. For example, consider a bidder facing an incredibly strong opponent that will price them out of the auction no matter how they behave in early rounds. Since they cannot impact their own utility, they can play any strategy in equilibrium, ranging from immediately dropping out of the auction to raising their opponent's prices up to their own value; their arbitrary choice will impact many auction outcomes of interest (e.g., auction length, revenue). We therefore advocate for adding small, "secondary" rewards to the RL environment, biasing agents toward certain strategies (e.g., preferring to drop out than to stay in) when all else is equal.

Pure or mixed equilibria.

An analyst must decide whether to focus solely on pure equilibria or also to consider mixed equilibria. There are several advantages to studying pure equilibria. First, working with pure equilibria is computationally beneficial: restricting to pure strategies greatly reduces the strategy space, tending to make it easier to train, validate, and evaluate policies. Second, some MARL algorithms may not be able to converge to mixed equilibria at all. For a broad class of MARL algorithms, the only stable points of the learning dynamics are pure equilibria [Flokas et al., 2020]; we believe that a similar result is likely to hold for many algorithms, such as CFR. While there is no guarantee that every game will have any pure equilibria, in our own experiments, we have found that MARL algorithms can find a pure equilibrium in the vast majority of our games.

We recommend considering whether it is sufficient to study pure equilibria. If there is reason to suspect that some interesting bidder behavior can only be represented with mixed equilibria (e.g., randomized bluffing), it is necessary to be cautious about the choice of MARL algorithm. Otherwise, we recommend enforcing that policies play pure strategies by rounding them to the nearest pure strategy,

deterministically playing the action with the highest probability in the policy; we refer to such rounded policies as *modal*. This rounding step is necessary for finding pure strategy equilibria with MARL algorithms that play actions with probabilities that approach zero, but do so exceedingly slowly, potentially never reaching zero. One such example is PPO, which rewards entropy in its loss function.

Avoiding brittle equilibria.

Prior work in MARL has found that trained policies can fail catastrophically when facing new opponents. For example, in the cooperative card game Hanabi [Bard et al., 2020], policies trained through self-play can achieve near-perfect scores, but fail to get even a single point when playing with policies from other training runs. One way this can happen is when MARL algorithms identify *brittle* equilibria, in which players rely on perfect coordination with each other, which they achieve through their training history. Such equilibria may have no robustness to even minor misspecifications. Furthermore, in the domain of spectrum auctions, achieving such coordination would likely require illegal collusion between the bidders. To combat this problem, during training we recommend having policies *tremble* during training, deviating from the policy and playing an action uniformly at random with a small probability (e.g., 1%). Such trembling pushes agents away from equilibria that demand perfect coordination. Additionally, it serves as a way of further breaking down indifferences: even if two actions give the same rewards on-path, they may give different rewards off-path.

6.2.3 Validating and Interpreting Policies

Lastly, we discuss how to proceed after training a set of policies with a MARL algorithm.

Assessing convergence.

It is important to first understand whether a MARL algorithm has converged to an equilibrium: if not, it is difficult to know whether trends in bidders' strategies are simply caused by poor convergence. The ideal measure of convergence is the *NashConv* of the policy, which is defined as the sum of each agent's *regret*—the util-

ity gain they would achieve by playing a best-response, holding their opponents' strategies fixed. A NashConv of zero thus corresponds to a Nash Equilibrium, and a positive NashConv corresponds to an ϵ-Nash equilibrium, where ϵ is no greater than the NashConv. Unfortunately, NashConv is nontrivial to compute, as it requires requires finding the optimal solution to a single-agent RL problem for each player, a task that requires an exhaustive search. Thus, while it would be ideal to checkpoint the trained policy periodically and to analyze only the checkpoint with the smallest NashConv (which need not be the latest checkpoint), this is infeasible on all but the smallest of games.

When it is not possible to compute NashConv exactly, it can still be valuable to approximate it by computing approximate best responses. One way is to restrict the action space of the game to a smaller set of heuristics (such as in Section 6.2.1) and compute the best response on this restricted game. Another is to run a traditional single-agent RL algorithm, which has no guarantee of finding the optimal policy, but may nonetheless find a good policy. These approximate best responses give a lower bound on a player's regret: if one strategy would have increased their utility, then their best response must too. While this approach cannot guarantee that a strategy profile is an equilibrium, it can still be useful to have some evidence that each player's strategy is not easy to improve upon.

Note that it is dramatically easier to compute NashConv of a pure strategy profile, as this exponentially reduces the number of subtrees that must be examined. We recommend attempting to compute NashConv exactly if analyzing pure strategies. When one expects a pure strategy Nash equilibrium, it can also be useful to track the entropy of players' policies, which must decay to 0.

Interpreting multiple equilibria.

We have already stated some refinements above that rule out certain equilibria. Then, if a single equilibrium remains, one can simply proceed to study its strategy profile or check how its outcomes are affected by changes to the auction. However, a likely outcome is to wind up with a set of equilibria, making the analysis murkier: should an analyst treat one equilibrium as more important or plausible than another?

Without any other desiderata, we believe it is difficult to rank equilibria by their plausibility or frequency, as the exact choice of MARL algorithm biases which equilibria are likely to be found in ways that are difficult to predict. When faced with multiple equilibria, we advocate that no equilibrium is inherently more plausible than others. Thus, we recommend looking at the *range* of possible strategies or outcomes represented by the set of equilibria, rather than looking at e.g., the mean outcome across all samples, or only studying a single sample.

6.3 Case Study: Bid Processing in Clock Auctions

We now demonstrate our methodology on clock auctions, a modern iterative combinatorial auction format. As a case study, we focus on a specific question about the design of this auction: in which order should the auctioneer process simultaneous bids? We discuss how to instantiate each part of our methodology to tackle this question.

6.3.1 Clock Auctions

Clock auctions are an iterative combinatorial auction format that has been used to allocate spectrum, e.g., in the FCC's Auction 102 [FCC, 2019b] and the Canadian 3500 MHz auction [ISED, 2021]. Here, we describe a simplified auction format that draws heavily upon these real spectrum auctions; we provide a formal description of the auction in Appendix B.2.

In a clock auction for allocating spectrum, an auctioneer sells indivisible radio licenses to a number of bidders, with each bidder representing a telecom company. The licenses are divided into products, with each product corresponding to an equivalence class of geographic region and spectrum quality; there are typically several units of supply available for each product. The auctioneer sets an opening price for each product. Then, the auction proceeds over a series of rounds. In each round, bidders submit a vector-valued bid representing the number of units of each product they want to buy at the current prices. If any product is overdemanded—that is, has more aggregate demand than supply—then the auctioneer increases the price on each overdemanded product, informs all bidders of every product's aggregate demand and new price, and proceeds to the next round of the auction.

Otherwise, the auction ends, with each bidder winning the units they bid for, paying their final prices.

We highlight two additional rules that add considerable complexity to clock auction analysis. First, to discourage strategic bidding, bidders cannot bid for arbitrary bundles in each round. Future bids are constrained by past bids through an *activity* rule. We consider a simple implementation which assigns each bundle a number of "eligibility points" and requires bids to be weakly decreasing in eligibility points. Second, to avoid leaving licenses unsold, the auctioneer rejects bids that would lower the demand for a product below its supply. Combined with the activity rule, this creates a potential exposure problem, as bids may only be partially processed if points from a dropped product are needed to pick up a new product.

We now use our methodology to evaluate a potential design choice: *when multiple bidders simultaneously request to drop licenses, how should the auctioneer determine which bids to process first?* Processing every bidder's drop request could change a product from being over- to under-demanded, which the auction rules disallow. To avoid leaving licenses unsold, the auctioneer must instead process the drop requests sequentially. In practice (e.g., in the Canadian 3500 MHz auction), these ties are handled by randomly ordering the queue of requests and processing entire drop requests in this order, letting bidders drop as many licenses as they requested, until no excess demand remains. This "drop-by-bidder" mechanism can have high variance, processing either all or none of each bidder's submitted bid. We compare this mechanism to an alternative where a request to drop several licenses of a product is represented in the queue as multiple requests to drop an *individual license*. This alternative "drop-by-license" mechanism reduces variance, making it most likely that each bidder will have some of their drop requests processed. These bid processing mechanisms are well-suited for our computational methodology because they require a large amount of case-based reasoning to study, making them overwhelming to analyze with traditional methods but ideal for reinforcement learning.

6.3.2 Defining the Environment

In the remainder of this section, we describe how we instantiated the methodology to compare these two auction designs. We implemented our clock auction game in the framework of OpenSpiel [Lanctot et al., 2019].

Auction setup.

We consider auctions with two bidders and a single geographic region with two products: an unencumbered[3] product with one license of supply, and an encumbered product with four licenses, each of which only gives 60% of the bandwidth of an unencumbered license. Each license has eligibility points and an opening price proportional to the amount of bandwidth available. We focus on a specific state midway through the auction[4]: we assume that each bidder has previously bid for three of the encumbered licenses for several rounds, making it more expensive per MHz of spectrum than the unencumbered license. The bidders have a strategic choice to make: continue bidding on the more expensive product, or attempt to switch to the cheaper one (e.g., dropping two encumbered licenses to pick up one unencumbered license). The bid processing mechanism is used if both bidders attempt to switch. In the drop-by-bidder mechanism, the auctioneer chooses a random order over the bidders and processes all of one bidders' drops in order, allowing one random bidder to switch while the other's bid is unprocessed. In the drop-by-license mechanism, they instead process the individual requested drops in a random order, most often ending with both bidders dropping one encumbered license; in this case, the activity rule prevents both bidders from picking up the unencumbered license.[5]

We set the opening start-of-round prices to $12 million on the unencumbered product and $7 million on the encumbered product. We set the clock speed—the

[3] In spectrum auctions, certain licenses are *encumbered*, only covering a fraction of the MHz-Pop that a full, unencumbered license would. Encumbered licenses are usually priced at a discount to reflect their poorer coverage.

[4] We do not assume this particular midway point is part of any equilibrium of the game starting from the first round. We are interested in equilibrium behavior starting from this point.

[5] Real auctions often include a *grace period* rule, giving bidders a chance to regain lost activity in the following round. Our model does not include a grace period. However, notice that in this strategic setting, if both players lose activity, the auction is guaranteed to end, and a grace period rule would have no effect in these cases.

price increase after each round—to 5% of the current price. Thus, after two rounds of bidding for the encumbered product, its price increased by two clock increments to $7.72 million.

Bidder utilities.

We assume that each bidder has a finite number of types, with each type being equally likely. Each type has a quasilinear utility function, meaning that they have a value for each bundle, and their utility for winning a bundle is its value minus its price paid. We choose these values by drawing them from the MRVM value model [Weiss et al., 2017], informed by the 2014 Canadian spectrum auction. Within a region, MRVM models bidders as having a critical amount of spectrum which they need in order to provide a high quality of service, and low marginal value when they have much more or less than this critical amount. Bidders are thus described by two parameters: their *value per subscriber*, the amount of value they would gain from winning all of the spectrum, and their *market share*, the fraction of spectrum which they need to obtain half of this maximum value. Their values are then sigmoidal with respect to the fraction of spectrum they win, with high marginal values within $\pm 15\%$ of their market share, and low marginal values outside of this range.[6] For each type, we draw their value per subscriber uniformly at random between $20 million and $30 million, and their market share uniformly at random between 35% and 50%. Note that these non-linear value functions can make "switch" bids risky, as bidders can lose a large amount of value if a switch bid is only partially processed.

In preliminary experiments, we found that some samples from this value model led to games that were either strategically uninteresting (e.g., had a bidder too weak to have a chance of winning a single license) or had game trees too large to be feasible (e.g., could last 20 rounds or more). To guard against these problems, we ran rejection sampling, resampling values that would exhibit one of these problems. We generated values for bidders having 1, 2, 3, 5, and 7 types, generating 5 value profiles for each number of types. We provide the precise details of the rejection sampling process and an example of bidders' value functions in Appendix B.3.

[6] We note a similar sigmoidal modeling choice in [Bichler et al., 2022], which studies the FCC's C-band.

Secondary rewards.

With no further changes, these auctions can admit many different actions that lead to the same terminal rewards: for example, attempting to drop from a demand of 3 to a demand of 0 often has the same effect as dropping to a demand of 1. We address this problem by adding secondary rewards (as described in Section 6.2.2). Specifically, each round, we subtract a penalty between zero and one from each player's utility. A penalty of zero corresponds to choosing the most profitable bundle at the current prices, and a penalty of one to the least profitable; other bundles' penalties interpolate between these two extremes. These secondary rewards can be understood as giving bidders an incentive to prefer straightforward bids and shorter auctions in the absence of any other differences, breaking equalities between actions' terminal rewards. These penalties are removed after training, and all of our reported NashConv values are with respect to the unmodified game.

6.3.3 Finding Equilibria

We ran two MARL algorithms: a Monte Carlo sampling variant of Counterfactual Regret Minimization, and Proximal Policy Optimization, an on-policy deep reinforcement learning method.

Monte Carlo Counterfactual Regret Minimization

Monte Carlo Counterfactual Regret Minimization (MCCFR) is a tabular MARL algorithm. It is effectively a generalization of regret matching to imperfect information extensive form games: at each information state, it tracks a *regret* for each action, playing actions with probabilities proportional to their regret. Specifically, we use a variant known as External Sampling MCCFR [Lanctot et al., 2009], which avoids traversing the entire game tree by randomly sampling chance outcomes and opponent actions. Initial experimentation revealed that two common performance tricks were helpful: we use the regret matching plus [Tammelin, 2014] algorithm instead of regret matching, and we use linear CFR [Brown and Sandholm, 2019b], which discounts regrets over time so that later iterations have larger impacts. Lastly, as is common for CFR, we analyze the average strategy across all iterations, rather than the last iterate.

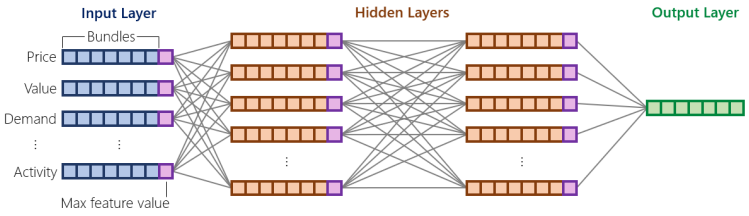


Figure 6.1: The AuctionNet neural network architecture.

We make one modification to this standard algorithm. To help steer MCCFR away from brittle equilibria, we train against *trembling* opponents. Specifically, when we sample an opponent's action, we assume that they tremble 1% of the time, picking an action uniformly at random rather than using their trained policy.[7] This trembling only occurs during training; there is no trembling in our final policies.

Proximal Policy Optimization

Our second algorithm is a deep reinforcement learning algorithm, Proximal Policy Optimization (PPO) [Schulman et al., 2017]. PPO is an on-policy algorithm inspired by classic policy gradient methods. It represents its policy with a neural network; roughly, to train this policy, it collects a sample of information states, actions, and their eventual rewards, then updates its policy to increase the probability it places on actions leading to high rewards. PPO differs from a standard policy gradient method in that its objective function penalizes large changes in action probability, causing it to change its policy more slowly during training.

In order to use PPO, we must choose a neural network architecture. We experiment with two. The first is a feed-forward neural network, accepting a flat vector describing all variables known in the information state as input. While this is a simple architecture, it is unlikely to be effective: such a network would have to learn how to process facts about every single bundle of licenses independently. To address this issue, we also test an equivariant architecture [Hartford et al., 2018], which is shown in Figure 6.1. In this architecture, each bundle has an associated set of features, and the network must perform the same computation on every bun-

[7] We do not make the training player tremble, as external sampling MCCFR already evaluates every one of their own actions rather than sampling just one.

dle. Comparisons can be made across bundles using max-pooling. This enforces that the network must apply the same function to each bundle, giving it a strong inductive bias toward learning a single, simple function and applying it globally.

For both MCCFR and PPO, we convert all trained policies to pure strategies, placing probability 1 on the modal action. This conversion is computationally important, making it possible to compute NashConv by pruning all actions with probability 0.

6.3.4 Validating and Interpreting Policies

We compute several metrics about each of our trained policies. For each simulation, we record the auction revenue, welfare, length (in rounds), the number of unsold licenses, and number of lotteries in bid processing. The latter is relevant to the design decision being studied as it shines light on whether agents are engaging in strategies that trigger lotteries or carefully avoiding them. We validate the convergence of our modal policies by computing NashConv with a standard depth-first search algorithm.

We also compare our results to simulations of straightforward bidders. Concretely, we define a *straightforward bidder* as a bidder who, in each round, bids for the bundle of licenses that would give the highest utility at the current prices. [8] This is a natural definition of straightforward bidding behavior as it is myopic, with no regard to price increases or activity constraints in future rounds. It is also analogous to existing definitions of straightforward bidding in simultaneous multi-round auctions [e.g., Milgrom, 2004], a related auction format.

6.4 Experimental Results

We now discuss three sets of experiments and their results. First, we tuned our two MARL algorithms. We found configurations that could identify policies with low

[8] While we refer to this strategy as straightforward, we note that it is far from straightforward to define straightforward bidding in this auction. Any bid that involves dropping licenses might be rejected. This is especially important when part—but not enough—of a drop bid goes through and a bidder does not have enough activity to pick up what they intended. A bidder may therefore not want to place bids that could leave them exposed. Reasoning about bids in this manner requires a more complex demand forecasting model.

(often zero!) NashConv in our games. Second, we ran our tuned algorithms on games with two different bid processing algorithms. We observed that the modification to bid processing yields substantially different auction outcomes due to non-trivial changes in bidder behavior. Third, we tested how well these algorithms scaled to larger games. We found that they often still converged to approximate equilibria.

Our computational environment is described in Appendix B.1. Except where noted otherwise, we ran our MARL algorithms for one hour of walltime.

6.4.1 Ablations and Hyperparameter Tuning

First, we experiment with various hyperparameter settings for each of our MARL algorithms.

MCCFR: Trembling Opponents and Secondary Rewards

MCCFR has few hyperparameters that need to be tuned. Here, we evaluate our recommendations from Sections 6.2.2 and 6.2.2, where we advocated for breaking ties between rewards for actions, and for having opponents tremble to avoid brittle strategies. To test these techniques, we compared four variants of MCCFR, varying whether opponents trembled during training and whether the game included secondary rewards. We ran each variant with 10 random seeds on 10 different five-type games (comprising of 5 valuation samples and 2 bid processing rules), making for a total of 100 runs for each variant. For each run, we computed NashConv at six checkpoints, after 100, 200, 500, 1000, 2000, and 3000 iterations, respectively. Runs took between 10 and 60 minutes.

The results are shown in Figure 6.2. All four variants consistently converged to approximate equilibria. However, there were consistent differences between some of the variants: training with secondary rewards led to lower NashConv early in training, and more frequently converged to exact Nash equilibria. Training against trembling opponents had a more varied impact on convergence, increasing NashConv early in training but having no effect at later checkpoints. In the remainder of our experiments, we opted to run MCCFR with both of these features, reasoning that they were likely to steer MCCFR away from brittle equilibria and lead to faster

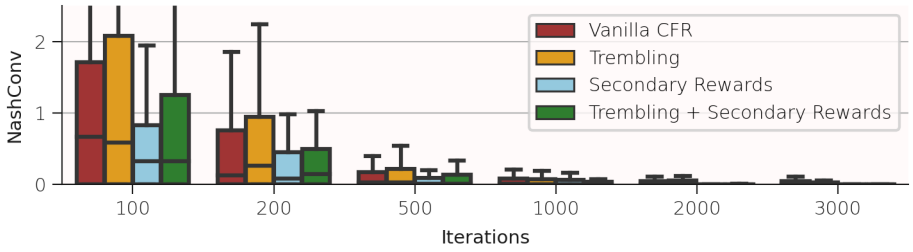


Figure 6.2: NashConv of MCCFR ablations, varying secondary rewards and trembling opponents.

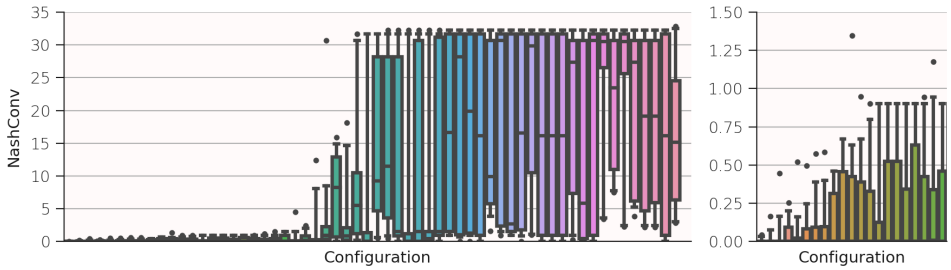


Figure 6.3: NashConv of PPO hyperparameter tuning runs, showing all 60 configurations (left) and a detailed view of the 20 best configurations (right). Configurations are sorted by 95th percentile of NashConv. Whiskers denote 5th and 95th percentiles.

convergence.

PPO: Architecture and Hyperparameter Optimization

Compared to CFR, PPO has many hyperparameters and its performance is highly sensitive to these hyperparameter settings. These hyperparameters control, among other elements, its neural network architecture, loss function, and optimizers. We tuned these hyperparameters using random search. Specifically, we sampled 60 random configurations of these hyperparameters and ran each with 3 random seeds on 6 different five-type games (comprising of 3 valuation samples and 2 bid processing rules), making for a total of 18 runs for each configuration. For each, we ran PPO for 30 minutes, computing NashConv once at the end. We provide details about the hyperparameters and our random sampling distributions in Appendix B.4.

The results are shown in Figure 6.3. The hyperparameter configurations varied substantially in their ability to find approximate equilibria, with many frequently producing policies with very high NashConv. However, a few configurations consistently converged to approximate equilibria; the best configuration always returned policies with a NashConv below 0.1. Notably, this configuration selected the AuctionNet architecture (as do 8 of the best 10 configurations). We used this single best PPO configuration in the remainder of our experiments.

6.4.2 Effects of Bid Processing Algorithms

We now turn to our main economic question: *how does the auction's bid processing algorithm affect bidders' strategies and outcomes of the auction?* To answer this question, we experimented with bidder valuations with 1, 2, 3, 5, and 7 types, generating 5 samples of each for a total of 25 value profiles.[9] For each value profile, we created two games: one with each bid processing algorithm. These games varied in size, with the 1-type games having 10–100 information states, and the 7-type games having 500-700 information states. We ran MCCFR and PPO on each game with 10 random seeds, for a total of 1000 MARL algorithm runs. We discarded 24 runs that had a NashConv greater than 0.1, leaving 976 runs; 904 of these had a NashConv of 0, meaning that they converged to Nash equilibria.

Our results are shown in Figure 6.4. This plot shows four metrics: the length of the auction, number of lotteries (i.e., the number of rounds where the bid processing algorithm can produce two or more different outcomes), revenue, and welfare. These metrics were averaged over ten thousand episodes to account for randomness in bidders' assigned types and bid processing outcomes. Each point represents an (approximate) equilibrium found by one of the two MARL algorithms. We provide supplementary plots separating the equilibria found by each of the two MARL algorithms in Appendix B.5.

Our main finding is that the two bid processing rules lead to qualitatively different outcomes: when bidders have 3 or more types, equilibria under the drop-by-license rule can produce longer auctions with higher revenue and lower welfare

[9] Bidders' valuations were sampled independently for each of these games. This means that, for example, bidders' values in the 1-type games were not necessarily equal to any values in games with more than 1 type.

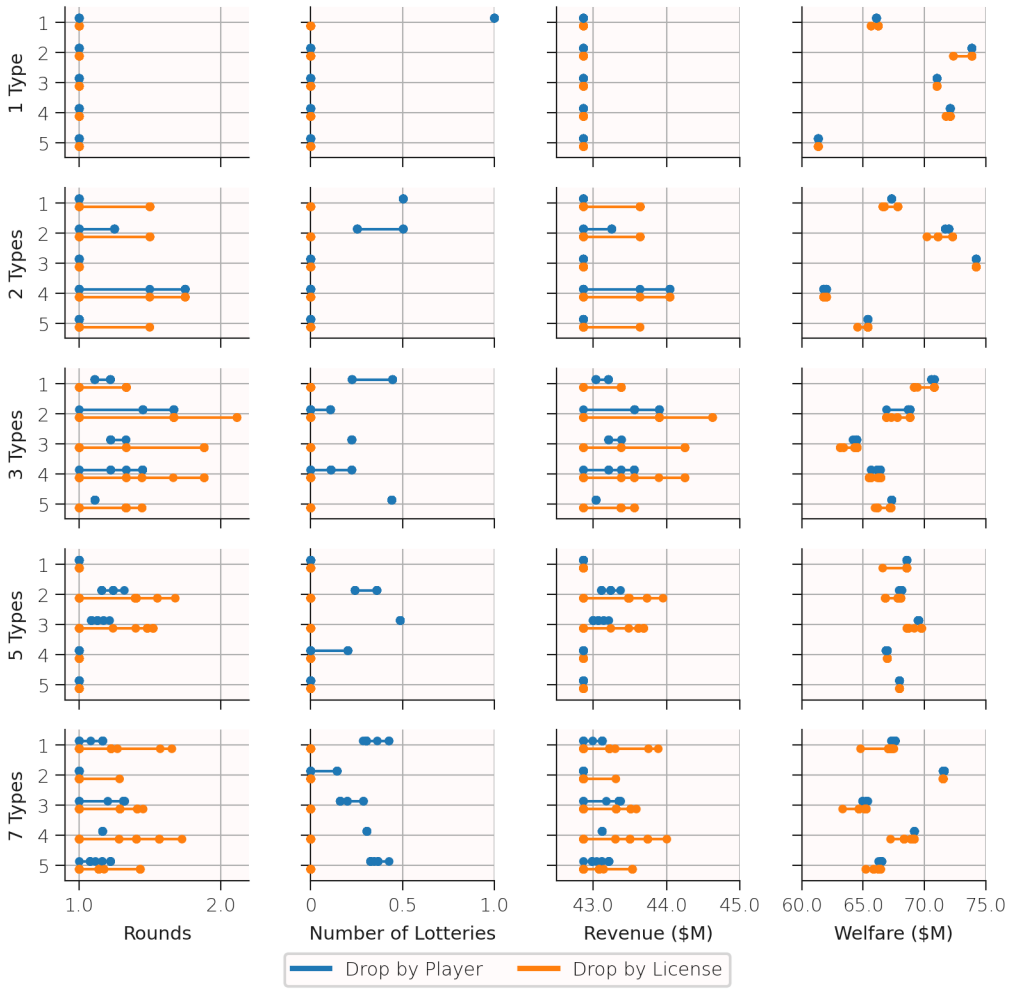


Figure 6.4: Auction outcomes using MCCFR and PPO on 2-player games with 1 to 7 types.

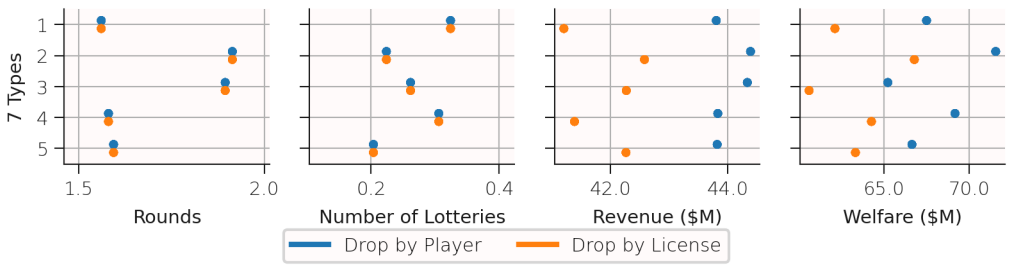


Figure 6.5: Auction outcomes under straightforward bidding on 2-player games with 7 types.

than under the drop-by-player rule. These differences in auction outcomes are caused by differences in bidder behavior. Under drop-by-player, when both bidders attempt to switch their demand from the encumbered product to the unencumbered one, the bid processing algorithm can result in a lottery; when this happens, the auction immediately ends. Under drop-by-license, however, these lotteries are less desirable to bidders, as they risk dropping a single encumbered license which will not create enough activity to buy an unencumbered license. Bidders adjust their strategies accordingly, taking care to avoid simultaneously switching their demand, causing every single equilibrium under this rule to have no lotteries. In many games, this adjustment can lead to longer auctions, creating additional revenue as the prices increase.

Notably, the differences between the two bid processing algorithms vanish when bidders only have a single type. In these games, bidders have complete information about their opponent's values (and, in equilibrium, their opponent's strategy). They are then able to avoid lotteries regardless of the bid processing algorithm in most games, leading to identical auction lengths and revenues under the two rules. The results are also ambiguous with 2 types per bidder, where no equilibrium leads to a bid processing lottery in 3 of our 5 games. These findings highlight the importance of modelling the environment as a Bayesian game with multiple types for each player, as doing so can lead to qualitatively different findings. Lastly, these findings also highlight the importance of searching for multiple equilibria; if we had only sampled a single equilibrium from each game, we would have drawn very different conclusions.

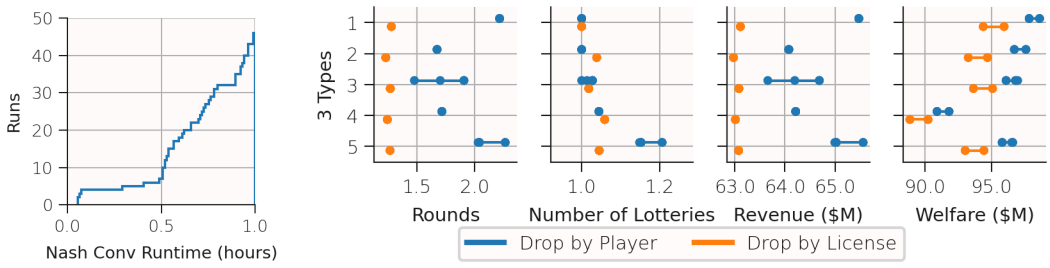
We also simulated the auctions under straightforward bidding. Straightforward simulation results for 7-type games are shown in Figure 6.5, with additional plots for the other games in Appendix B.5. Figure 6.5 shows that ignoring bidders' incentives would lead to entirely different conclusions. Here, both rules lead to auctions of the same length, as bidders act the same way regardless of the bid processing algorithm. Furthermore, it is now the drop-by-player rule that leads to more revenue: under the drop-by-license rule, it is common for a license to be left unsold, losing revenue. The drop-by-player rule also produces higher welfare for a similar reason, as it is inefficient to leave licenses unsold. We note that straightforward bidding is indeed not an equilibrium in these auctions: on the 7-type games, NashConv ranges from 0.18 to 3.10.

6.4.3 Scaling to Larger Games

Encouraged by how well our algorithms performed on the games above, we created a set of larger games to test how well our MARL algorithms would scale. To do so, we added a third bidder and a fifth encumbered license. To keep the games tractable, we also increased the clock increment from 5% to 20%. We initialized the auction in a similar state as before, having each bidder bid for three encumbered licenses for two rounds. We generated 5 value profiles with 3 types per bidder, yielding a total of 10 games across the two bid processing algorithms. These games are substantially larger than the 2-player games: the smallest have around 8,000 information states, while the largest have over 40,000. We ran MCCFR for 8 hours on each game.

Validating trained policies on these larger games was substantially more difficult. At the end of each run, we attempted to compute NashConv for up to 1 hour. Within this time limit, we were only able to compute NashConv for 46 of the 100 runs; the distribution of runtimes, plotted in Figure 6.6a, shows that NashConv took at least 30 minutes on most runs. Nonetheless, when we were able to compute NashConv, it was low. The worst run's NashConv was 0.03 and 36 runs had NashConvs of 0. Thus, it is possible for MCCFR to find equilibria of these larger games.

For completeness, Figure 6.6b shows the auction metrics for all 100 runs, in-



(a) (b)

Figure 6.6: (a) NashConv runtime distribution; (b) auction outcomes using MCCFR on 3-player games.

cluding those for which we could not compute NashConv. It is interesting that the differences between the rules in this setting are quite unlike those in the 2-player games, with the drop-by-player rule leading to longer auctions, higher revenue, and higher welfare. One potential explanation is that under the drop-by-player rule, all three players can safely attempt to switch their demand from the encumbered product to the unencumbered product, allowing for further competition in the following round.

6.5 Conclusions and Discussion

Multiagent reinforcement learning algorithms are rapidly maturing, having recently reached several significant milestones. This chapter argues that—if games are modelled appropriately and the right algorithmic choices are made—MARL can be used to analyze iterative combinatorial auctions that resist both pen-and-paper and computational Bayes–Nash analysis. We laid out a methodology for modelling an auction for use with MARL analysis, finding near-equilibria using MARL algorithms, and both validating and interpreting the results. We then offered a case study on bid processing in clock auctions, showing that MARL analysis could reach an economically meaningful conclusion in a setting that would not have admitted equilibrium analysis, and for which a simulation-based analysis would have yielded the wrong conclusion, and finding evidence that MARL analysis has the potential to scale to even larger settings.

We hope that we have persuaded the reader that MARL has something to offer auction analysis; however, our own work has clearly only scratched the surface. We thus conclude with an in-depth discussion of useful next steps: future clock auction design choices that would be fruitful to analyze; promising avenues for improving our methodology; and open theoretical questions.

6.5.1 Promising Auction Design Questions

As we described in Section 6.3.1, an auction designer faces many potential choices about how to instantiate each aspect of an auction, each of which could influence bidding behavior and ultimately outcomes. Here, we describe three such design choices that we believe would be amenable to MARL analysis, demonstrating the relevance of each with citations either to the academic literature or to bidder consultations with policymakers. For each, we discuss the design choice's strategic implications and how it might impact revenue, welfare, and/or length of auctions.

Clock speed A designer chooses the exponential increment by which prices of overdemanded products increase. Larger increments allow for quicker auctions, but reduce opportunities for price discovery as there is less feedback for bidders. Bidders react to these increments; in a consultation for the 3800 MHz Canadian auction, Telus "argued that increments [of 10%-20%] would create an accelerated cadence at the start of the auction that may prove contrary to the desired intention of promoting price discovery" [ISED, 2022]. An auctioneer may even consider dynamic schemes, such as modifying the clock speed partway through the auction, or using a separate clock speed for each product, with steeper price increases for more overdemanded products.

Information disclosure At the end of each round, the auctioneer discloses information about the current state of the auction. Commonly, this includes the prices and units of aggregate demand for each product. An auctioneer could vary this policy—for example, only disclosing whether each product was over- or underdemanded—hoping to improve outcomes by disincentivizing strategic bidding. An opposite extreme is to deanonymize all bids. Such a suggestion was made in a

consultation for the 3500 MHz Canadian auction: "Eastlink opposed the use of anonymous bidding, stating that it generally favours the large national incumbent bidders as they inherently have more information available to them during the price discovery rounds" [ISED, 2021].

Activity rules There is a body of work on how to select activity rules, mostly centered around which one to pick from the perspectives of discouraging strategic manipulations and minimizing complexity for the bidder. For example, Harsha et al. [2010] define *strong activity rules* as those which admit straightforward bidding strategies and exclude strategies that not consistent with straightforward bidding under some utility function. Ausubel and Baranov [2020] propose that an activity rule should (1) ensure demand and price move in opposite directions, (2) always allow straightforward bidding, and (3) always allow repeating the previous bid; the authors prove that an activity rule based on the Generalized Axiom of Revealed Preference (GARP) uniquely satisfies the three axioms under non-linear prices. Of course, neither of these papers is backed by equilibrium analysis, as this would be intractable; it would thus be valuable to use MARL analysis to estimate the extent to which these activity rules allow strategic manipulation.

It would also be valuable to study bidder behavior under a variety of value models—for example, modelling risk-averse bidders, or spiteful bidders who gain utility from competitors paying high prices—to understand the extent to which any findings are robust to the choice of bidder model.

6.5.2 Methodological Improvements

Enumerating many equilibria.

Our methodology recommended identifying multiple equilibria by running MARL algorithms with different random seeds. This approach is simple to implement, but may not be able to find all equilibria of a game; furthermore, inductive biases in the algorithm almost certainly make it more likely to find some equilibria than others. While this may be seen as a good thing if one believes that equilibria that learning dynamics struggles to find are less plausible, it may also be seen as a blind spot.

An alternate approach that could be worth investigating is termed *quality diversity* in single-agent RL [e.g., Pugh et al., 2016]; it focuses on finding multiple distinct, good policies. Future work should develop and evaluate algorithms to increase the diversity of equilibria, expanding our ability to fully understand the equilibria of an auction.

Interpreting bidding behavior.

Our analysis in our case study focused mostly on outcomes. Particularly for bidder-focused "playbook" applications, it can be important to derive insights about what constitutes a good strategy. This requires that learned policies not only be measurable but be interpretable. Bertsimas and Paskov [2022] use CFR to train poker agents and then fit decision trees to the CFR strategies through supervised learning to produce interpretable poker strategies. Future work could similarly focus on distilling bidding behavior. A set of composable bidding strategy building blocks could be valuable both for understanding auctions and as heuristics for scale.

6.5.3 Theoretical Questions

Convergence to Nash equilibrium.

In two-player zero-sum games, CFR is guaranteed to converge to an approximate Nash equilibrium. However, auctions are not zero-sum, and existing guarantees for general-sum games are far weaker. Morrill et al. [2021] show that CFR is only guaranteed to converge to a counterfactual coarse correlated equilibrum, a much weaker solution concept. Despite this lack of theoretical justification, on our auctions, we found that MCCFR consistently converged to pure Bayes–Nash (near-)equilibrium. Is there a class of games (larger than two-player zero-sum) in which existing MARL algorithms converge to Nash equilibria? Understanding the space of such games would help with selecting an appropriate MARL algorithm.

Alternative solution concepts.

While we have focused on finding (Bayes–)Nash equilibria, this solution concept can admit unrealistic behavior. One concrete problem is that it is possible for

our learned policies to make non-credible threats in off-path states. For example, a bidder could adopt a trigger strategy, bidding round after round to increase prices to unprofitable heights if their opponent does not coordinate with them in earlier rounds. These strategies are unrealistic, as no reasonable bidder would go bankrupt to punish an opponent. Various equilibrium refinements seek to eliminate this behavior, such as sequential equilibrium [Kreps and Wilson, 1982]; developing MARL algorithms that converge to these refinements would be a valuable direction for future work.

Bibliography

K. I. Aardal, S. P. Van Hoesel, A. M. Koster, C. Mannino, and A. Sassano. Models and solution techniques for frequency assignment problems. *Annals of Operations Research*, 153(1):79–129, 2007. → page 40

D. J. Abraham, A. Blum, and T. Sandholm. Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. In *Proceedings of the 8th ACM conference on Electronic Commerce*, EC '07, pages 295–304, New York, NY, USA, 2007. Association for Computing Machinery. → page 1

M. Abramowicz. Modeling settlement bargaining with algorithmic game theory. Research Paper 2020-80, GWU Legal Studies, 2020. → page 113

G. Adomavicius, S. P. Curley, A. Gupta, and P. Sanyal. Effect of information feedback on bidder behavior in continuous combinatorial auctions. *Management Science*, 58:811–830, 2012. → page 60

N. Agarwal and P. Somaini. Demand analysis using strategic reports: An application to a school choice mechanism. *Econometrica*, 86(2):391–444, 2018. → page 61

J. C. Aker. Information from markets near and far: Mobile phones and agricultural markets in Niger. *American Economic Journal: Applied Economics*, 2(3):46–59, 2010. → page 7

M. Allman, I. Ashlagi, I. Lo, J. Love, K. Mentzer, L. Ruiz-Setz, and H. O'Connell. Designing school choice for diversity in the San Francisco unified school district. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, page 290–291, New York, NY, USA, 2022. Association for Computing Machinery. → pages 1, 59

G. Anandalingam, R. W. Day, and S. Raghavan. The landscape of electronic market design. *Management Science*, 51:316–327, 2005. → page 59

C. Ansótegui, M. Sellmann, and K. Tierney. A gender-based genetic algorithm for the automatic configuration of algorithms. In *Conference on Principles and Practice of Constraint Programming (CP)*, pages 142–157, 2009. ISBN 3-642-04243-0, 978-3-642-04243-0. → pages 39, 53

T. W. Anthony, T. Eccles, A. Tacchetti, J. Kram'ar, I. M. Gemp, T. C. Hudson, N. Porcel, M. Lanctot, J. P'erolat, R. Everett, S. Singh, T. Graepel, and Y. Bachrach. Learning to play no-press Diplomacy with best response policy iteration. *ArXiv*, abs/2006.04635, 2020. → page 113

I. Asimov. Runaround. In *Astounding Science Fiction*, 1942. → page 94

S. Athey and D. Nekipelov. A structural model of sponsored search advertising auctions. In *6th Ad Auctions Workshop*, volume 15, 2010. → page 61

S. Athey, J. Levin, and E. Seira. Comparing open and sealed bid auctions: Evidence from timber auctions. *The Quarterly Journal of Economics*, 126(1): 207–257, 2011. → page 61

L. M. Ausubel. An efficient ascending-bid auction for multiple objects. *American Economic Review*, 94(5):1452–1475, December 2004. → page 111

L. M. Ausubel and O. Baranov. Revealed preference and activity rules in dynamic auctions. *International Economic Review*, 61(2):471–502, 2020. → page 140

L. M. Ausubel, P. Cramton, and P. Milgrom. The clock-proxy auction: A practical combinatorial auction design. In *Handbook of spectrum auction design*, pages 120–140. MIT, 2006. → page 111

L. M. Ausubel, C. Aperjis, and O. Baranov. Market design and the FCC Incentive Auction. Working Paper, 2017. → pages 62, 76, 79

K. Back and J. F. Zender. Auctions of divisible goods with endogenous supply. *Economics Letters*, 73(1):29–34, 2001. ISSN 0165-1765. → page 65

A. Bakhtin, N. Brown, E. Dinan, G. Farina, C. Flaherty, D. Fried, A. Goff, J. Gray, H. Hu, A. P. Jacob, M. Komeili, K. Konath, M. Kwon, A. Lerer, M. Lewis, A. H. Miller, S. Mitts, A. Renduchintala, S. Roller, D. Rowe, W. Shi, J. Spisak, A. Wei, D. J. Wu, H. Zhang, and M. Zijlstra. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378:1067 – 1074, 2022. → page 113

T. Balyo, N. Froleyks, M. Heule, M. Iser, M. Järvisalo, and M. Suda, editors. *Proceedings of SAT Competition 2020: Solver and Benchmark Descriptions*.

Department of Computer Science Report Series B. Department of Computer Science, University of Helsinki, 2020. → page 182

M. Banchio and A. Skrzypacz. Artificial intelligence and auction design. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, page 30–31, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391504. → page 113

T. W. Bank. Mobile cellular subscriptions (per 100 people). https://data.worldbank.org/indicator/IT.CEL.SETS.P2, 2018. Accessed: 2018-03-02. → page 7

N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, I. Dunning, S. Mourad, H. Larochelle, M. G. Bellemare, and M. Bowling. The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280:103216, 2020. ISSN 0004-3702. → page 123

C. Bazelon. Good enough? *Available at SSRN 4188984*, 2022. → page 62

L. F. Bergquist. Pass-through, competition, and entry in agricultural markets: Experimental evidence from Kenya. 2017. → page 7

L. F. Bergquist, C. McIntosh, and M. Startz. Search costs, intermediation, and trade: Experimental evidence from Ugandan agricultural markets. *CEGA Working Paper Series No. WPS-173*, 2021. → pages 8, 9, 28, 30, 31, 32

D. Bertsimas and A. Paskov. World-class interpretable poker. *Machine Learning*, 111(8):3063–3083, 2022. → page 141

M. Bichler, V. Fux, and J. K. Goeree. Designing combinatorial exchanges for the reallocation of resource rights. *Proceedings of the National Academy of Sciences*, 116(3):786–791, 2019. → pages 1, 59

M. Bichler, M. Fichtl, S. Heidekrüger, N. Kohring, and P. Sutterer. Learning equilibria in symmetric auction games using artificial neural networks. *Nature Machine Intelligence*, 3(8):687–695, 2021. → page 113

M. Bichler, P. R. Milgrom, and G. Schwarz. Taming the communication and computation complexity of combinatorial auctions: The FUEL bid language. *Management Science*, 69:2217–2238, 2022. → pages 60, 128

M. Bichler, M. Fichtl, and M. Oberlechner. Computing Bayes–Nash equilibrium strategies in auction games via simultaneous online dual averaging. *Operations Research*, 2023. → page 113

S. Bikhchandani, S. De Vries, J. Schummer, and R. V. Vohra. An ascending Vickrey auction for selling bases of a matroid. *Operations research*, 59(2): 400–413, 2011. → page 60

V. Bosshard and S. Seuken. The cost of simple bidding in combinatorial auctions. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, EC '21, page 157, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450385541. URL https://doi.org/10.1145/3465456.3467609. → pages 112, 117

V. Bosshard, B. Bünz, B. Lubin, and S. Seuken. Computing bayes-nash equilibria in combinatorial auctions with verification. *Journal of Artificial Intelligence Research*, 69:531–570, 2020. → page 96

M. Bowling, N. Burch, M. Johanson, and O. Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, 2015. → page 113

N. Brown and T. Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018. → page 113

N. Brown and T. Sandholm. Superhuman AI for multiplayer poker. *Science*, 365 (6456):885–890, 2019a. ISSN 0036-8075. → page 113

N. Brown and T. Sandholm. Solving imperfect-information games via discounted regret minimization. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019b. ISBN 978-1-57735-809-1. → page 129

J. Bulow and P. Klemperer. Auctions versus negotiations. *The American Economic Review*, 86(1):180–194, 1996. ISSN 00028282. → page 65

C. Calsamiglia, C. Fu, and M. Güell. Structural estimation of a model of school choices: The Boston mechanism versus its alternatives. *Journal of Political Economy*, 128(2):642–680, 2020. → page 61

A. Camacho and E. Conover. The impact of receiving price and climate information in the agricultural sector. *IDB Working Paper Series, No. IDB-WP-220*, 2011. → page 9

I. Caragiannis, C. Kaklamanis, P. Kanellopoulos, and M. Kyropoulou. On the efficiency of equilibria in generalized second price auctions. In *Proceedings of*

the 12th ACM Conference on Electronic Commerce, EC '11, page 81–90, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450302616. → page 106

M. Cary, A. Das, B. Edelman, I. Giotis, K. Heimerl, A. R. Karlin, C. Mathieu, and M. Schwarz. Greedy bidding strategies for keyword auctions. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, EC '07, page 262–271, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595936530. → page 112

T. N. Cason, K. N. Kannan, and R. B. Siebert. An experimental study of information revelation policies in sequential auctions. *Management Science*, 57:667–688, 2011. → page 60

H. Chung, D. Freund, and D. B. Shmoys. Bike angels: An analysis of Citi Bike's incentive program. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, COMPASS '18, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450358163. → page 1

D. A. Cohen, M. C. Cooper, G. Escamocher, and S. Živnỳ. Variable and value elimination in binary constraint satisfaction via forbidden patterns. *Journal of Computer and System Sciences*, 81(7):1127–1143, 2015. ISSN 0022-0000. → page 51

P. Cramton, H. López, D. Malec, and P. Sujarittanonta. Design of the reverse auction in the FCC Incentive Auction. 2015. → page 62

G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006. URL https://igraph.org. → page 28

O. David-West. Esoko networks: facilitating agriculture through technology. GIM case study no. b061, 2010. → page 9

R. W. Day and S. Raghavan. Fair payments for efficient allocations in public sector combinatorial auctions. *Management Science*, 53:1389–1406, 2007. → page 60

D. Delacrétaz, S. D. Kominers, and A. Teytelboym. Matching mechanisms for refugee resettlement. *American Economic Review*, 113(10):2689–2717, October 2023. → pages 1, 59

L. Di Gaspero and A. Schaerf. Easysyn++: A tool for automatic synthesis of stochastic local search algorithms. In *Conference on Engineering Stochastic Local Search Algorithms (SLS)*, pages 177–181, 2007. ISBN 978-3-540-74445-0. → page 39

S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016. → page 168

Django Software Foundation. Django. URL https://djangoproject.com. → page 28

A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pages 3071–3076, 2013. → page 168

U. Doraszelski, K. Seim, M. Sinkinson, and P. Wang. Ownership concentration and strategic supply reduction. Technical Report 23034, National Bureau of Economic Research, January 2017. → pages 62, 67, 68

Q. Duong, Y. Vorobeychik, S. Singh, and M. P. Wellman. Learning graphical game models. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, IJCAI'09, page 116–121, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc. → page 95

L. Einav, C. Farronato, and J. Levin. Peer-to-peer markets. *Annual Review of Economics*, 8:615–635, 2016. → page 10

R. Engelbrecht-Wiggans and C. M. Kahn. Low-revenue equilibria in simultaneous ascending-bid auctions. *Management Science*, 51:508–518, 2005. → page 59

M. Fafchamps and B. Minten. Relationships and traders in Madagascar. *The Journal of Development Studies*, 35(6):1–35, 1999. → page 26

M. Fafchamps and B. Minten. Impact of SMS-based agricultural information on Indian farmers. *The World Bank Economic Review*, 26(3):383–414, 2012. → page 9

Farm Africa. Our work in Uganda. https://www.farmafrica.org/uganda/uganda, 2018. Accessed: 2018-03-02. → page 6

Farmgain Africa. Farmgain Africa. http://farmgainafrica.org/, 2018. Accessed: 2018-03-02. → page 9

FCC. In the matter of expanding the economic and innovation opportunities of spectrum through incentive auctions. *Notice of Proposed Rulemaking*, FCC 12–118, 10 2012. → page 64

FCC. Office of engineering and technology releases and seeks comment on updated OET-69 software. *FCC Public Notice*, DA 13-138, February 2013. → page 41

FCC. Information related to Incentive Auction repacking feasibility checker. *FCC Public Notice*, DA 14-3, 1 2014. → page 55

FCC. Application procedures for broadcast Incentive Auction scheduled to begin on March 29, 2016. *FCC Public Notice*, DA 15-1183, 10 2015a. → pages 42, 82

FCC. Repacking constraint files. http://data.FCC.gov/download/incentive-auctions/Constraint_Files/, 2015b. Accessed 2015-11-20. → page 41

FCC. Reverse auction opening prices. http://wireless.FCC.gov/auctions/incentive-auctions/Reverse_Auction_Opening_Prices_111215.xlsx, 2015c. Accessed 2015-11-15. → page 170

FCC. Incentive Auction: Reverse auction - bids. https://auctiondata.FCC.gov/public/projects/1000/reports/reverse-bids, 2019a. Accessed 2019-11-28. → page 70

FCC. Auction 102 clock phase technical guide. 2019b. → page 125

J. A. Ferejohn, R. Forsythe, and R. G. Noll. An experimental analysis of decision making procedures for discrete public goods: A case study of a problem in institutional design. *V. L. Smith, ed. Research in Experimental Economics*, pages 1–58, 1979. → page 60

B. A. Ferguson and P. Milgrom. Market design for surface water. Working Paper 32010, National Bureau of Economic Research, December 2023. → page 59

L. Flokas, E. V. Vlatakis-Gkaragkounis, T. Lianeas, P. Mertikopoulos, and G. Piliouras. No-regret learning and mixed Nash equilibria: they do not mix. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546. → page 122

A. Fréchette, N. Newman, and K. Leyton-Brown. Solving the station repacking problem. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 702–709. AAAI Press, 2016. → pages 53, 71, 84

D. Freund, S. G. Henderson, and D. B. Shmoys. Minimizing multimodular functions and allocating capacity in bike-sharing systems. *Production and Operations Management*, 27(12):2346–2349, 2018. → pages 1, 59

M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub. clasp: A conflict-driven answer set solver. In *Logic Programming and Nonmonotonic Reasoning*, pages 260–265. 2007. → page 50

C. P. Gomes and B. Selman. Algorithm portfolios. *Artificial Intelligence*, 126(1): 43–62, 2001. → page 53

Y. A. Gonczarowski, O. Heffetz, and C. Thomas. Strategyproofness-exposing mechanism descriptions. *ArXiv*, abs/2209.13148, 2022. → page 4

S. Govindan and R. Wilson. A global Newton method to compute Nash equilibria. *J. Economic Theory*, 110:65–86, 2003. → page 107

J. R. Green and J.-J. Laffont. Incentives in public decision making. 1979. → page 66

F. Gul and E. Stacchetti. Walrasian Equilibrium with Gross Substitutes. *Journal of Economic Theory*, 87(1):95–124, July 1999. → page 23

J. C. Harsanyi and R. Selten. A general theory of equilibrium selection in games. MIT Press, 1988. → page 112

P. Harsha, C. Barnhart, D. Parkes, and H. Zhang. Strong activity rules for iterative combinatorial auctions. *Computers & Operations Research*, 37:1271–1284, 07 2010. → page 140

J. Hartford, D. Graham, K. Leyton-Brown, and S. Ravanbakhsh. Deep models of interactions across sets. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1909–1918. PMLR, 2018. → page 130

N. Hildebrandt, Y. Nyarko, G. Romagnoli, and E. Soldani. Price information, inter-village networks, and "bargaining spillovers": Experimental evidence from Ghana. *Working Paper*, 2020. → page 9

H. H. Hoos. Programming by optimization. *Communications of the ACM*, 55(2): 70–80, Feb. 2012. ISSN 0001-0782. → page 39

F. Hutter, H. H. Hoos, K. Leyton-Brown, and T. Stützle. ParamILS: an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36(1):267–306, 2009. → pages 39, 53

F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization Conference (LION)*, pages 507–523, 2011. → pages 39, 53

F. Hutter, M. Lindauer, S. Bayless, H. Hoos, and K. Leyton-Brown. Results of the configurable SAT solver challenge 2014. Technical report, University of Freiburg, Department of Computer Science, 2014a. → page 53

F. Hutter, M. López-Ibáñez, C. Fawcett, M. Lindauer, H. H. Hoos, K. Leyton-Brown, and T. Stützle. AClib: A benchmark library for algorithm configuration. In *Conference on Learning and Intelligent Optimization (LION)*, pages 36–40. Springer, 2014b. → page 50

Infotrade. Infotrade - market information services. http://www.infotradeuganda.com/, 2018. Accessed: 2018-03-02. → page 9

ISED. Consultation on a licensing framework for mobile broadband services (MBS) — 700 MHz band. *Spectrum Management and Telecommunications*, 2012. Accessed 2024-02-12. → page 111

ISED. Technical, policy and licensing framework for spectrum in the 600 MHz band. *Spectrum Management and Telecommunications*, SLPB-002-18, 2018. Accessed 2024-02-12. → page 111

ISED. Policy and licensing framework for spectrum in the 3500 MHz band. *Spectrum Management and Telecommunications*, SLPB-001-20, 4 2021. Accessed 2024-02-12. → pages 111, 125, 140

ISED. Policy and licensing framework for spectrum in the 3800 MHz band. *Spectrum Management and Telecommunications*, SPB-002-22, 06 2022. Accessed 2024-02-12. → pages 111, 139

M. Järvisalo, D. Le Berre, O. Roussel, and L. Simon. The international SAT solver competitions. *Artificial Intelligence Magazine*, 33(1):89–92, 2012. → page 50

R. Jensen. The digital provide: Information (technology), market performance, and welfare in the South Indian fisheries sector. *The Quarterly Journal of Economics*, 122(3):879–924, 2007. → page 7

A. X. Jiang and K. Leyton-Brown. A polynomial-time algorithm for Action-Graph Games. In *American Association for Artificial Intelligence (AAAI)*, pages 679–684, 2006. → page 108

A. X. Jiang and K. Leyton-Brown. Bayesian action-graph games. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1*, NIPS'10, page 991–999, Red Hook, NY, USA, 2010. Curran Associates Inc. → page 97

A. X. Jiang, K. Leyton-Brown, and A. Pfeffer. Temporal action-graph games: a new representation for dynamic games. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, page 268–276, Arlington, Virginia, USA, 2009. AUAI Press. ISBN 9780974903958. → page 109

A. X. Jiang, K. Leyton-Brown, and N. A. Bhat. Action-graph games. *Games and Economic Behavior*, 71(1):141–173, 2011. → pages 94, 95, 107, 108

S. Kadioglu, Y. Malitsky, M. Sellmann, and K. Tierney. ISAC–instance-specific algorithm configuration. In *European Conference on Artificial Intelligence (ECAI)*, pages 751–756, 2010. ISBN 978-1-60750-605-8. → page 39

M. Kearns and L. Dworkin. A Computational Study of Feasible Repackings in the FCC Incentive Auctions. *arXiv:1406.4837 [cs]*, June 2014. arXiv: 1406.4837. → page 61

M. Kearns, M. L. Littman, and S. Singh. Graphical models for game theory. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, UAI'01, page 253–260, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558608001. → page 94

F. P. Kelly and R. Steinberg. A combinatorial auction with multiple winners for universal service. *Management Science*, 46:586–596, 2000. → page 60

A. R. KhudaBukhsh, L. Xu, H. H. Hoos, and K. Leyton-Brown. SATenstein: Automatically building local search SAT solvers from components. *Artificial Intelligence*, 232:20–42, 2016. ISSN 0004-3702. → pages 39, 50

J. L. Kiddoo, E. Kwerel, S. Javid, M. Dunford, G. M. Epstein, C. E. Meisch Jr, K. L. Hoffman, B. B. Smith, A. B. Coudert, R. K. Sultana, et al. Operations research enables auction to repurpose television spectrum for next-generation wireless technologies. *INFORMS Journal on Applied Analytics*, 49(1):7–22, 2019. → pages 61, 92

R. Knutson and T. Gryta. Verizon, AT&T May Face Bidding Limits in Spectrum Auction. *Wall Street Journal*, Apr. 2014. ISSN 0099-9660. URL http://www.wsj.com/articles/SB10001424052702304626304579510154106120342. http://www.wsj.com/articles/SB10001424052702304626304579510154106120342, Accessed 2016-12-12. → page 37

D. Koller and B. Milch. Multi-agent influence diagrams for representing and solving games. *Games and Economic Behavior*, 45:181–221, 2003. → page 109

S. D. Kominers, A. Teytelboym, and V. P. Crawford. An invitation to market design. *Oxford Review of Economic Policy*, 33(4):541–571, 2017. ISSN 0266-903X. → page 10

E. Krasnokutskaya and K. Seim. Bid preference programs and participation in highway procurement auctions. *American Economic Review*, 101(6): 2653–2686, 2011. → page 61

D. M. Kreps and R. Wilson. Sequential equilibria. *Econometrica*, 50(4):863–894, 1982. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1912767. → page 142

A. M. Kwasnica, J. O. Ledyard, D. Porter, and C. DeMartini. A new and improved design for multiobject iterative auctions. *Management Science*, 51: 419–434, 2005. → page 60

M. Lanctot, K. Waugh, M. Zinkevich, and M. Bowling. Monte Carlo sampling for regret minimization in extensive games. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, NIPS'09, page 1078–1086, Red Hook, NY, USA, 2009. Curran Associates Inc. ISBN 9781615679119. → pages 121, 129

M. Lanctot, E. Lockhart, J.-B. Lespiau, V. Zambaldi, S. Upadhyay, J. Pérolat, S. Srinivasan, F. Timbers, K. Tuyls, S. Omidshafiei, D. Hennes, D. Morrill, P. Muller, T. Ewalds, R. Faulkner, J. Kramár, B. D. Vylder, B. Saeta, J. Bradbury, D. Ding, S. Borgeaud, M. Lai, J. Schrittwieser, T. Anthony, E. Hughes, I. Danihelka, and J. Ryan-Davis. OpenSpiel: A framework for reinforcement learning in games. *CoRR*, abs/1908.09453, 2019. → page 127

J. D. Levin and A. Skrzypacz. Properties of the combinatorial clock auction. *The American Economic Review*, 106:2528–2551, 2016. → page 112

K. Leyton-Brown. The viability of exact feasibility checking. Accessed June 11, 2017. Talk at the Stanford Institute for Economic Policy Research Conference on the design of the U.S. Incentive Auction for reallocating spectrum between wireless telecommunications and television broadcasting, February 2013. URL https://www.cs.ubc.ca/~kevinlb/talks/2013-Feasibility-Testing.pdf. → page 85

K. Leyton-Brown, P. Milgrom, and I. Segal. Economics and computer science of a radio spectrum reallocation. *Proceedings of the National Academy of Sciences*, 114(28):7202–7209, 2017. ISSN 0027-8424. → pages 1, 59

S. Li. Obviously strategy-proof mechanisms. *American Economic Review*, 107 (11):3257–87, 2017. → pages 60, 63, 70

Z. Li, M. Lanctot, K. R. McKee, L. Marris, I. M. Gemp, D. Hennes, P. Muller, K. Larson, Y. Bachrach, and M. P. Wellman. Combining tree-search, generative models, and Nash bargaining concepts in game-theoretic reinforcement learning. *ArXiv*, abs/2302.00797, 2023. → page 113

M. López-Ibáñez, J. Dubois-Lacoste, L. Pérez Cáceres, T. Stützle, and M. Birattari. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58, 2016. → pages 39, 53

P. Milgrom. *Putting Auction Theory to Work*. Churchill Lectures in Economics. Cambridge University Press, 2004. ISBN 9781139449168. → pages 3, 36, 111, 131

P. Milgrom and I. Segal. Clock auctions and radio spectrum reallocation. *Journal of Political Economy*, 128(1):1–31, 2020. → pages 60, 66, 90

P. R. Milgrom and S. Tadelis. How artificial intelligence and machine learning can impact market design. Technical report, National Bureau of Economic Research, 2018. → page 10

S. Mitra, D. Mookherjee, M. Torero, and S. Visaria. Asymmetric information and middleman margins: An experiment with Indian potato farmers. *The Review of Economics and Statistics*, 100(1):1–13, 03 2018. ISSN 0034-6535. → page 9

J.-N. Monette, Y. Deville, and P. Van Hentenryck. *Aeon: Synthesizing Scheduling Algorithms from High-Level Models*, volume 47, pages 43–59. 01 2009. ISBN 978-0-387-88842-2. → page 39

D. Morrill, R. D'Orazio, R. Sarfati, M. Lanctot, J. R. Wright, A. R. Greenwald, and M. Bowling. Hindsight and sequential rationality of correlated play. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5584–5594, 2021. → page 141

R. B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6 (1):58–73, 1981. → pages 65, 82

R. B. Myerson and M. A. Satterthwaite. Efficient mechanisms for bilateral trading. *Journal of Economic Theory*, 29(2):265 – 281, 1983. ISSN 0022-0531. → page 24

E. A. Nakasone. The role of price information in agricultural markets: Evidence from rural Peru. *Working Paper*, 2014. → page 9

N. Newman, A. Fréchette, and K. Leyton-Brown. Deep optimization for spectrum repacking. *Communications of the ACM*, 61(1):97–104, 2017. → page 84

N. Newman, L. F. Bergquist, N. Immorlica, K. Leyton-Brown, B. Lucier, C. McIntosh, J. Quinn, and R. Ssekibuule. Designing and evolving an electronic agricultural marketplace in Uganda. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, pages 1–11, 2018. → page 6

N. Newman, K. Leyton-Brown, P. Milgrom, and I. Segal. Incentive auction design alternatives: A simulation study. 2024. → page 112

Nielsen. DMA regions. https://www.nielsen.com/us/en/contact-us/intl-campaigns/dma-maps/, 2022. Accessed 2022-03-06. → page 169

E. Nudelman, K. Leyton-Brown, G. Andrew, C. Gomes, J. McFadden, B. Selman, and Y. Shoham. Satzilla 0.9. Solver description, International SAT Competition, 2003. → page 53

OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017. → pages 18, 28

A. Pacaud, A. Bechler, and M. Coupechoux. Bidding efficiently in simultaneous ascending auctions with budget and eligibility constraints using simultaneous move Monte Carlo tree search, 2023. → page 113

R. Paes Leme, V. Syrgkanis, and E. Tardos. Sequential auctions and externalities. In *SODA*, 2011. → page 109

I. Palacios-Huerta, D. C. Parkes, and R. Steinberg. Combinatorial auctions in practice. *SSRN Electronic Journal*, 2021. → page 111

A. S. Pekec and M. H. Rothkopf. Combinatorial auction design. *Management Science*, 49:1485–1503, 2003. → page 59

J. Pérolat, B. D. Vylder, D. Hennes, E. Tarassov, F. Strub, V. de Boer, P. Muller, J. T. Connor, N. Burch, T. W. Anthony, S. McAleer, R. Élie, S. H. Cen, Z. Wang, A. Gruslys, A. Malysheva, M. Khan, S. Ozair, F. Timbers, T. Pohlen, T. Eccles, M. Rowland, M. Lanctot, J.-B. Lespiau, B. Piot, S. Omidshafiei, E. Lockhart, L. Sifre, N. Beauguerlange, R. Munos, D. Silver, S. Singh, D. Hassabis, and K. Tuyls. Mastering the game of Stratego with model-free multiagent reinforcement learning. *Science*, 378:990 – 996, 2022. → page 112

T. Perrier, B. DeRenzi, and R. Anderson. USSD: The third universal app. In *Proceedings of the 2015 Annual Symposium on Computing for Development*, pages 13–21. ACM, 2015. → page 15

F. R. Pieroth, N. Kohring, and M. Bichler. Equilibrium computation in multi-stage auctions and contests. *ArXiv*, abs/2312.11751, 2023. URL https://api.semanticscholar.org/CorpusID:266362174. → page 113

R. Porter, E. Nudelman, and Y. Shoham. Simple search methods for finding a Nash equilibrium. *Games and Economic Behavior*, 63:642–662, 2008. → page 108

J. K. Pugh, L. B. Soros, and K. O. Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016. → page 141

Z. Rabinovich, V. Naroditskiy, E. H. Gerding, and N. R. Jennings. Computing pure Bayesian-Nash equilibria in games with finite actions and continuous types. *Artificial Intelligence*, 195:106–139, 2013. ISSN 0004-3702. → pages 96, 108, 112

S. J. Rassenti, V. L. Smith, and R. L. Bulfin. A combinatorial auction mechanism for airport time slot allocation. *The Bell Journal of Economics*, 13(2):402–417, 1982. ISSN 0361915X. → page 60

F. Riedel and E. Wolfstetter. Immediate demand reduction in simultaneous ascending-bid auctions: a uniqueness result. *Economic Theory*, 29(3):721–726, 2006. → page 112

A. E. Roth. The economist as engineer: Game theory, experimentation, and computation as tools for design economics. *Econometrica*, 70(4):1341–1378, 2002. → page 1

M. H. Rothkopf, A. S. Pekec, and R. M. Harstad. Computationally manageable combinational auctions. *Management Science*, 44:1131–1147, 1998. → page 66

T. Roughgarden and C. Papadimitriou. Computing correlated equilibria in multi-player games. *Journal of the ACM*, 37:49–56, 2008. → page 94

T. Roughgarden and E. Tardos. Do externalities degrade GSP's efficiency? In *Workshop on Advertising Auctions*, 2012. → page 106

C. T. Ryan, A. X. Jiang, and K. Leyton-Brown. Computing pure strategy Nash equilibria in compact symmetric games. In *EC*, 2010. → page 109

T. Sandholm. Approaches to winner determination in combinatorial auctions. *Decision Support Systems*, 28(1):165–176, 2000. ISSN 0167-9236. → page 66

N. Santos, P. Tubertini, A. Viana, and J. P. Pedroso. Kidney exchange simulation and optimization. *Journal of the Operational Research Society*, 68(12): 1521–1532, 2017. → pages 1, 59

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. → page 130

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016. → page 112

D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. P. Lillicrap, K. Simonyan, and D. Hassabis. Mastering Chess and Shogi by self-play with a general reinforcement learning algorithm. *ArXiv*, abs/1712.01815, 2017. → page 112

N. J. Sitko and T. Jayne. Exploitative briefcase businessmen, parasites, and other myths and legends: Assembly traders and the performance of maize markets in eastern and southern Africa. *World Development*, 54:56 – 67, 2014. ISSN 0305-750X. → page 26

P. Somaini. Identification in auction models with interdependent costs. *Journal of Political Economy*, 128(10):3820–3871, 2020. → page 61

R. Ssekibuule. *Market Design for Agricultural Trade in Developing Countries*. PhD thesis, Radboud University Nijmegen, 2017. → page 9

R. Ssekibuule, J. A. Quinn, and K. Leyton-Brown. A mobile market for agricultural trade in Uganda. In *Proceedings of the 4th Annual Symposium on Computing for Development*, ACM DEV-4 '13, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450325585. → pages v, 7, 8, 9, 23

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. → page 121

O. Tammelin. Solving large imperfect information games using CFR+. *ArXiv*, abs/1407.5042, 2014. URL https://api.semanticscholar.org/CorpusID:9245873. → page 129

A. Teytelboym, S. Li, S. D. Kominers, M. Akbarpour, and P. Dworczak. Discovering auctions: Contributions of Paul Milgrom and Robert Wilson. *The Scandinavian Journal of Economics*, 123(3):709–750, 2021. → page 111

V. Thoma, V. Bosshard, and S. Seuken. Computing perfect Bayesian equilibria in sequential auctions. *ArXiv*, abs/2312.04516, 2023a. → page 113

V. Thoma, M. Curry, N. He, and S. Seuken. Learning best response policies in dynamic auctions via deep reinforcement learning. *ArXiv*, abs/2312.13232, 2023b. → page 113

D. Thompson, N. Newman, and K. Leyton-Brown. The positronic economist: a computational system for analyzing economic mechanisms. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 720–727. AAAI Press, 2017. → page 112

D. R. Thompson and K. Leyton-Brown. Revenue optimization in the generalized second-price auction. In *Proceedings of the Fourteenth ACM Conference on Electronic Commerce*, EC '13, page 837–852, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450319621. → page 94

D. R. Thompson, O. Lev, K. Leyton-Brown, and J. Rosenschein. Empirical analysis of plurality election equilibria. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS '13, page 391–398, Richland, SC, 2013. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450319935. → pages 94, 104

D. R. M. Thompson. *The positronic economist: a computational system for analyzing economic mechanisms*. PhD thesis, University of British Columbia, 2015. → page vii

D. R. M. Thompson and K. Leyton-Brown. Computational analysis of perfect-information position auctions. In *Proceedings of the 10th ACM Conference on Electronic Commerce*, EC '09, page 51–60, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584584. → pages 94, 104

D. R. M. Thompson and K. Leyton-Brown. Computational analysis of perfect-information position auctions. *Games and Economic Behavior*, 102: 583–623, 2017. ISSN 0899-8256. → page 112

D. R. M. Thompson, S. Leung, and K. Leyton-Brown. Computing Nash equilibria of action-graph games via support enumeration. In *Workshop on Internet and Network Economics*, 2011. → pages 95, 108

G. van der Laan, A. J. J. Talman, and L. van Der Heyden. Simplicial variable dimension algorithms for solving the nonlinear complementarity problem on a product of unit simplices using a general labelling. *Mathematics of Operations Research*, 12:377–397, 1987. → page 108

H. R. Varian. Position auctions. *International Journal of Industrial Organization*, 25:1163–1178, 2007. → page 104

Y. Vorobeychik and M. P. Wellman. Stochastic search methods for Nash equilibrium approximation in simulation-based games. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '08, page 1055–1062, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9780981738116. → page 96

Y. Vorobeychik, D. M. Reeves, and M. P. Wellman. Constrained automated mechanism design for infinite games of incomplete information. *Journal of Autonomous Agents and Multi-Agent Systems*, 25:313–351, 2012. → page 96

X. Wang, G. Q. Ma, A. Eden, C. Li, A. Trott, S. Zheng, and D. Parkes. Platform behavior under market shocks: A simulation framework and reinforcement-learning based study. In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 3592–3602, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450394161. → page 113

C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992. doi:10.1007/BF00992698. → page 121

M. Weiss, B. Lubin, and S. Seuken. SATS: A universal spectrum auction test suite. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 51–59, 2017. → pages 60, 115, 128

M. P. Wellman. Methods for empirical game-theoretic analysis. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, page 1552–1555. AAAI Press, 2006. ISBN 9781577352815. → page 117

S. J. Westfold and D. R. Smith. Synthesis of efficient constraint-satisfaction programs. *The Knowledge Engineering Review*, 16(1):69–84, March 2001. ISSN 0269-8889. → page 39

L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown. SATzilla: portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research*, 32: 565–606, June 2008. → page 53

L. Xu, H. H. Hoos, and K. Leyton-Brown. Hydra: automatically configuring algorithms for portfolio-based selection. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, page 210–216. AAAI Press, 2010. → pages 39, 54

R. Yang, B. Ford, M. Tambe, and A. Lemieux. Adaptive resource allocation for wildlife protection against illegal poachers. In *International Conference on Autonomous Agents and Multi-Agent Systems*, page 453–460, 2014. → pages 1, 59

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530, 2016. → page 39

S. Zheng, A. Trott, S. Srinivasa, D. C. Parkes, and R. Socher. The AI economist: Taxation policy design via two-level deep multiagent reinforcement learning. *Science Advances*, 8(18):eabk2607, 2022. → page 113

M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione. Regret minimization in games with incomplete information. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. → pages 113, 120, 121

Appendix A

Supporting Materials: Incentive Auction

A.1 FCC Band Plan

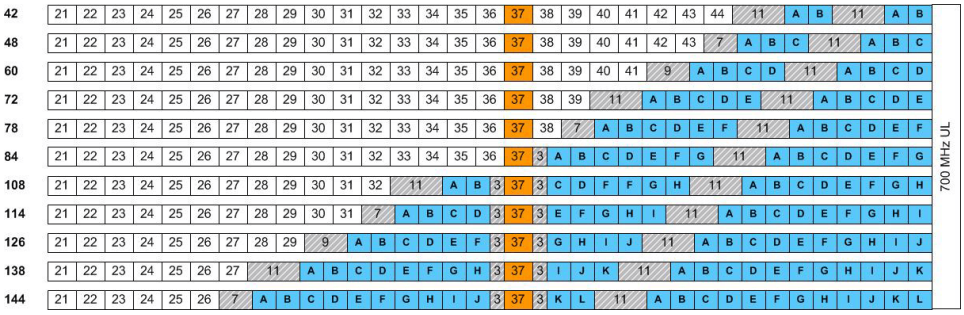


Figure A.1: The FCC's band plan for 600 MHz. Each row corresponds to a different clearing target and shows which channels would be repurposed and which would remain for TV broadcasting.

A.2 Reverse Auction Pseudocode

Algorithm 3 Multi-Stage UHF-Only Reverse Auction Without Impairment

Input: S_{exited} contains initial set of non-participating stations. S_{bidding} contains the set of participating stations. $\overline{c}$, $\mathcal{C}$, $\mathcal{I}$, $\mathcal{D}$ are the initial clearing target, set of channels, the interference constraints, and the station domains respectively. FC is the feasibility checker. p_0 is the starting clock price.

$S_{\text{winners}} \leftarrow \{\}$ ▷ Set of provisionally winning stations, initially empty
$\mathcal{P}(s) \leftarrow 0 \; \forall s \in S_{\text{bidding}}$ ▷ Provisionally winning prices; initially all 0
$\overline{\mathcal{C}} \leftarrow \{c \in \mathcal{C} \mid c < \overline{c}\}$ ▷ Channels available for repacking
$t \leftarrow 1$
while The incentive auction has not terminated **do**
  // Start a new stage
  $p_t \leftarrow p_0$ ▷ Reset clock prices
  while $|S_{\text{bidding}}| > 0$ **do**
    $p_t \leftarrow p_t - \max(0.05 \cdot p_t, 0.01 \cdot p_0)$ ▷ Decrease clock price
    // Check if any stations frozen in earlier stages have "caught up". Can be skipped in the first stage.
    for $\{s | s \in S_{\text{catchup}} \wedge p_t \cdot \text{SCORE}(s) \leq \mathcal{P}(s)\}$ **do**
      $S_{\text{catchup}} \leftarrow S_{\text{catchup}} \setminus \{s\}$
      if REPACKABLE($\{s\} \cup S_{\text{exited}}$) **then**
        $S_{\text{bidding}} \leftarrow S_{\text{bidding}} \cup \{s\}$
      else
        $S_{\text{winners}} \leftarrow S_{\text{winners}} \cup \{s\}$

    for $s \in S_{\text{bidding}}$ **do**
      $P_{s;t} \leftarrow p_t \cdot \text{SCORE}(s)$ ▷ Update prices
    Collect bid $b_{s;t}$ from each station in S_{bidding}
    for $s \in S_{\text{bidding}}$ **do** ▷ Bid processing
      if Not REPACKABLE($\{s\} \cup S_{\text{exited}}$) **then** ▷ s freezes
        $S_{\text{winners}} \leftarrow S_{\text{winners}} \cup \{s\}$
        $S_{\text{bidding}} \leftarrow S_{\text{bidding}} \setminus \{s\}$
        $\mathcal{P}(s) \leftarrow P_{s;t}$
      else if $b_{s;t} =$ Exit **then** ▷ s can be repacked and bid to exit
        $S_{\text{bidding}} \leftarrow S_{\text{bidding}} \setminus \{s\}$
        $S_{\text{exited}} \leftarrow S_{\text{bidding}} \cup \{s\}$
    $t \leftarrow t + 1$

  Conduct a forward auction for the cleared spectrum $\{c \in \mathcal{C} \mid c \geq \overline{c}\}$
  Reverse auction cost $\leftarrow \sum_{s \in S_{\text{winners}}} \mathcal{P}(s)$
  if Revenue raised in forward auction $\geq$ Reverse auction cost **then**
    Terminate the incentive auction
  else
    $\overline{c} \leftarrow$ next clearing target ▷ Selected according to band plan
    $\overline{\mathcal{C}} \leftarrow \{c \in \mathcal{C} \mid c < \overline{c}\}$ ▷ Update channels available for repacking
    $S_{\text{catchup}} \leftarrow \{s \in S_{\text{winners}} \mid \text{REPACKABLE}(\{s\} \cup S_{\text{exited}})\}$ ▷ Find stations that may unfreeze
    $S_{\text{winners}} \leftarrow S_{\text{winners}} \setminus S_{\text{catchup}}$
Pay $\mathcal{P}(s)$ to $s \in S_{\text{winners}}$ ▷ Incentive auction terminated

function REPACKABLE(S)
  Ask FC to find an assignment γ such that $\gamma(s) \in \mathcal{D}(s) \cap \overline{\mathcal{C}} \; \forall \, s \in S$, and $\gamma(s) = c \Rightarrow \gamma(s') \neq c'$ for all $\{(s, c), (s', c')\} \in \mathcal{I}$
  if A feasible assignment is found within cutoff **then**
    Return True
  else
    Return False

A.2.1 VHF Option Pricing

In this appendix, we describe how VHF options are priced in the reverse auction.

An Ideal Case: Similar Stations

The easiest case to reason about is when there are three stations which serve identical populations: a UHF station, an HVHF station, and an LVHF station, with the property that none of these stations can co-exist on the same band, and that if any station moves to a lower band, its interference constraints in the new band are identical to the station it displaces. There are then four ways to clear the UHF channel, each of which create the same amount of value and hence should cost the same amount.

1. The UHF station can go to OFF for a price of $\$X$
2. The UHF station can go LVHF for a price of $\$Y$ (where $\$Y < \$X$) and the LVHF station can go to OFF at a price of $\$X - \$Y$
3. The UHF station can move to HVHF for a price of $\$Z$ (where $\$Z < \$Y$) and the HVHF station can go OFF at a price of $\$X - \$Z$
4. The UHF station can move to HVHF for a price of $\$Z$, the LVHF station can go to OFF for a price of $\$X - \$Y$, and the HVHF station can go to LVHF at a price of $\$Y - \$Z$.

Benchmark Prices

The reverse auction uses *benchmark prices* based on the above thought experiment: the auction first pretends that we are in the scenario of the above example where comparable stations exist and computes a benchmark price which it later transforms into an actual price as described below.

To compute the benchmark price for a station s in round t for the option of moving into b, the previous round's benchmark price is decremented by a **fraction** of d_t. This fraction is called a *reduction coefficient*, $r_{s;t;b}$. We will describe the computation of reduction coefficients shortly.

$$p_{s;t;b} = p_{s;t-1;b} - r_{s;t;b} \cdot d_t \tag{A.1}$$

The initial benchmark prices, denoted $p_{0,b}$, are chosen prior to the auction (with $p_{s;0;\text{UHF}} = 0$). The benchmark price is then converted to an actual price as follows.

$$P_{s;t;b} = \text{score}\,(s) \cdot \max\left\{0, \min\left\{p_{s;t;\text{OFF}}, p_{s;t;b} - p_{s;t;v_{s,\text{pre}(s)}}\right\}\right\} \tag{A.2}$$

Let us break down this formula: The prices are weighted by volume as before. The $\max$ ensures the price is non-negative and the $\min$ upper bounds the price by the price offered to a UHF station to go to OFF. Lastly, the second term in the $\min$ reflects the pricing division described in Section A.2.1.

Vacancy

Before describing the calculation of reduction coefficients, we need to introduce *vacancy*. Vacancy, denoted $V_{t,s,b}$, is a heuristic that estimates the competition that a station s faces for a spot in band b in round t, with higher values indicating less competition. More formally, vacancy is a volume weighted average of a function f over potential competitors for the vacant space in the band: Let $G\,(t, s, b)$ denote the set containing both s and stations which have the possibility to interfere with s in b which are currently bidding on options below b. The function f is computed by taking the number of channels in b to which s can be feasibly assigned (given the current assignment of the exited stations) and normalizing by the number of channels in b. If s cannot be assigned to any channels, 0.5 is used in place of the numerator.

$$f\,(t, s, b) = \frac{\text{\# of feasible channels for } s \text{ in } b \text{ in round } t}{\text{\# of channels in } b} \tag{A.3}$$

$$V_{t,s,b} = \frac{\sum_{s' \in G(t,s,b)} \text{score}\,(s') \cdot f\,(t, s', b)}{\sum_{s' \in G(t,s,b)} \text{score}\,(s')} \tag{A.4}$$

Reduction Coefficients

We now provide equations for computing the reduction coefficients. The full decrement is always applied to going off-air, that is $r_{t,s,\text{OFF}} = 1$.

The reduction coefficient for moving to HVHF is:

$$r_{t,s,\text{HVHF}} = \frac{p_{0,\text{HVHF}} \cdot \sqrt{V_{t,s,\text{UHF}}}}{(p_{0,\text{OFF}} - p_{0,\text{HVHF}}) \cdot \sqrt{V_{t,s,\text{HVHF}}} + p_{0,\text{HVHF}} \cdot \sqrt{V_{t,s,\text{UHF}}}} \tag{A.5}$$

Finally, the reduction coefficient for moving to LVHF is:

$$r_{t,s,\text{LVHF}} = \left(\frac{(p_{0,\text{LVHF}} - p_{0,\text{HVHF}}) \cdot \sqrt{V_{t,s,\text{HVHF}}}}{(p_{0,\text{OFF}} - p_{0,\text{LVHF}}) \cdot \sqrt{V_{t,s,\text{LVHF}}} + (p_{0,\text{LVHF}} - p_{0,\text{HVHF}}) \cdot \sqrt{V_{t,s,\text{HVHF}}}} \right) \cdot (1 - r_{t,s,\text{HVHF}}) + r_{t,s,\text{HVHF}}$$

A.3 Additional Details of the BD Value Model

A.3.1 Inferring Bounds on Values from Bids

The FCC's data contains the selected band and price (and fallback band and price when applicable) for each bid that was processed. It additionally contains the offers and set of bands for which each station s accepted opening prices, which we call $\text{PermissibleStartBands}_s$.[1]

We began with the trivial bounds $0 \leq v_{s,\text{UHF}} \leq \infty$. We tightened bounds by applying the following rules[2] to the released bids:

1. $\text{OFF} \in \text{PermissibleStartBands}_s \implies v_{s,\text{UHF}} \leq P_{s;\text{OFF;Open}}$
2. $\text{OFF} \notin \text{PermissibleStartBands}_s$[3] $\implies v_{s,\text{UHF}} \geq P_{s;\text{OFF;Open}}$

[1] Prior to the auction, stations accepted or rejected prices for each of their eligible bands; the optimization that determined the initial starting assignment was then constrained to only place stations on accepted bands.

[2] One could imagine further tightening bounds based on stations' bids for their VHF options; however, for some stations, this leads to contradictory bounds. We therefore discarded any bidding information related to VHF, including all bids made after a station (possibly) moved to a VHF band.

[3] This includes both stations that were eligible to participate and chose not to, and those that participated conditional on starting in a VHF band.

3. $s \in S_{\text{winners}} \wedge \text{post}(s) = \text{OFF} \implies v_{s,\text{UHF}} \leq \mathcal{P}(s)$ [4]

4. $b_{s,t} = \text{OFF} \vee \text{fallback}_{s,t} = \text{OFF} \implies v_{s,\text{UHF}} \leq P_{s;\text{OFF};t}$

5. $(b_{s,t} = \text{Exit} \vee \text{fallback}_{s,t} = \text{Exit}) \wedge \gamma_t(s) = \text{OFF} \implies v_{s,\text{UHF}} \geq P_{s;\text{OFF};t}$

In words, we inferred that a station that included (did not include) starting off-air as a permissible option had a value less than (greater than) its opening price for starting off-air. A station that froze while bidding for off-air and became a winner had a home band value less than its compensation. Whenever a station bid to remain off-air, including as a fallback bid when attempting to move between bands, we inferred that the station's value was less than its price for remaining off-air. Similarly, whenever a station bid to drop out of the auction (including as a fallback bid) while off-air, we inferred that the station's value was greater than its price for remaining off-air.

A.3.2 Fitting our Value Distribution

Let x_s and y_s represent lower and upper bounds respectively on s's observed \$/pop sample n_s, i.e., $x_s = \frac{v_{s,\text{UHF,Lower Bound}}}{\text{Population}(s)}, y_s = \frac{v_{s,\text{UHF,Upper Bound}}}{\text{Population}(s)}$. Let Z be a list containing all of the x_s and y_s in ascending order, such that Z_1 is the smallest element and $Z_{2|S|}$ the largest. Our problem was then as follows.

$$\text{maximize} \prod_{s \in S} \big(N(y_s) - N(x_s)\big) \tag{A.6}$$

$$\text{subject to } N(x_s), N(y_s) \in [0,1] \, \forall s \in S \tag{A.7}$$

$$N(Z_1) \leq N(Z_2) \leq \cdots \leq N(Z_{2|S|}) \tag{A.8}$$

Noting that taking the log of the objective function preserves its maximum, we minimized the negative log likelihood, $-\sum_{s \in S} \log\left(N(y_s) - N(x_s)\right)$. This is a constrained optimization problem with a nonlinear convex objective function and convex constraints. The constraints ensure that the values of N must fall between 0 and 1 and that N must be non-decreasing. The solution uniquely identifies the

[4] We exclude station 35630, which received \$0 for its license.

value of N at each point x_s and y_s. We translated this problem into a second-order cone program using `cvxpy` [Diamond and Boyd, 2016] and solved it using the `ECOS` [Domahidi et al., 2013] solver.

We then fit a GPD to each of our tails. Specifically, we labelled the bottom 15% segment of the CDF the left tail and top 30% segment the right tail. The CDF of the GPD is $F(x) = 1 - (1 + \frac{x-\mu}{\sigma} \cdot \xi)^{\frac{-1}{\xi}}$ for $\xi \neq 0$, where μ, σ, ξ are the location, scale, and shape parameters respectively. We fit the left tail to the set of points Z falling in the region described (about 7% of our data). We did not have any data to fit the right tail of the distribution, so we fit a GPD using control points such that a \$/pop value 12.5% higher than any we observed occurred at $y = 0.975$ and one 25% higher than any we ever observed occurred at $y = 0.99$.

While we recognize that the value distribution's right tail is the most *ad hoc* part of our model, we believe that the choices we made about this part of the distribution are unlikely to have substantially impacted our results. The reason is that stations with sufficiently high values are unlikely to choose to participate even at the incentive auction's opening prices, at which point it does not matter to the auction's outcome exactly how high those values are. Indeed, we calculated that only 3.4% of UHF stations nationally would have *any* chance of participating in the auction conditional on their values having been sampled from any part of the right tail. Since each station has a 30% chance of having its value sampled from the right tail in any given simulation, in expectation any change to this section of the curve would alter the bidding behaviour of fewer than 1% of UHF stations. Furthermore, it is likely that conditional on a station's value having been drawn from the right tail and the station having chosen to participate, these stations would exit early in the auction before interference constraints start to bind tightly, and that this behaviour would not depend on the exact shape of the right-tail distribution.

A.4 Additional Simulation Details

A.4.1 Computational Environment

Our experiments were performed on two different compute clusters (the same cluster was used consistently for each experiment). Our first cluster ran on AWS, using

c5.18xlarge nodes. Each node had 72 vCPUs (each a hyperthread on a 3.0 GHz Intel Xeon Platinum 8000-series processor) and 144 GB of RAM. Our second cluster consisted of nodes equipped with 32 2.10GHz Intel Xeon E5-2683 v4 CPUs with 40960 KB cache and 96 GB RAM. SATFC runs a portfolio of 8 algorithms in parallel; jobs were always scheduled such that each simulation had access to 8 CPUs and 20 GB of RAM.

A.4.2 Handling Missing Data in the MCS Value Model

The MCS model does not provide values for stations in Hawaii, Puerto Rico, or the Virgin Islands. We therefore excluded all of these "non-mainland" stations from our analysis completely when using both value models to preserve the same interference graph across simulations. We note that there are few such stations and they reach relatively small populations, so this exclusion is unlikely to have impacted our qualitative findings.

The MCS model also does not provide values for 25 mainland UHF stations. For these stations, we set $v_{s,\text{UHF}}$ to be proportional to population. We sampled the constant of proportionality from a log-normal distribution, where the mean and variance were calculated from samples of $\frac{v_{s,\text{UHF}}}{\text{Population}(s)}$ from other stations in that station's Designated Market Area (DMA)[5] or nationally if there were no other stations in the DMA.

A.4.3 Additional Value Model Details

The Gaussian noise in our value model can occasionally cause a station to violate $v_{s,\text{UHF}} > v_{s,\text{HVHF}} > v_{s,\text{LVHF}}$. In these rare cases, we resample. We rounded all values to the nearest thousand dollars. In rare cases when the value model provides an extremely low sample for $v_{s,\text{UHF}}$, we set $v_{s,\text{UHF}} = \$3000$.

A.4.4 Initial Band Assignment

In the real auction, stations could choose to participate conditional on being initially assigned to one of a subset of bands. One role of the initial clearing target optimization was to accommodate such stations. Since we could not replicate this

[5] Nielsen divides the US into 210 geographic DMAs [Nielsen, 2022].

optimization, we did not allow stations to select their preferred starting bands. Instead, in our simulations, all participating stations start off-air. In the real incentive auction, 85% of stations indicated off-air as their preferred starting band and only 5% of stations declined off-air as a permissible starting band.

A.4.5 Canadian Stations

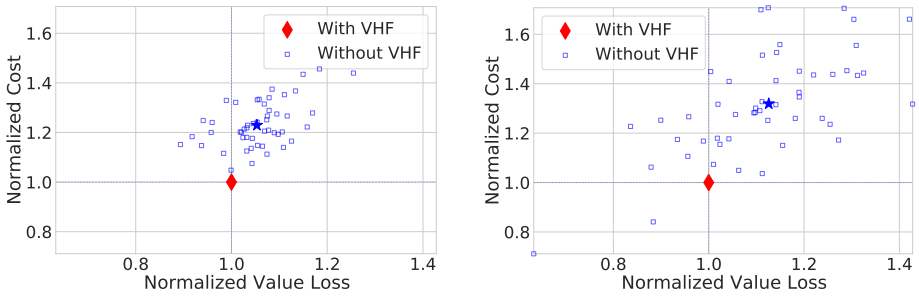
Canadian stations were also involved in the repacking process (they could be moved to a new channel, but did not participate in the auction and could not be purchased). We included all 793 Canadian stations in our simulations.

A.4.6 Station Volumes

We used population and interference values from the FCC's opening prices document [FCC, 2015c], noting that the links heuristic is calculated on the full interference graph. We continue to use these values in our UHF-only auctions, even though the underlying interference graph changes (the number of links goes down). We observe that the interference graph somewhat similarly changed between each stage of the incentive auction (the values correspond to the full graph, or the final stage of the band plan) but nevertheless the same values for the links heuristic were used throughout the auction.

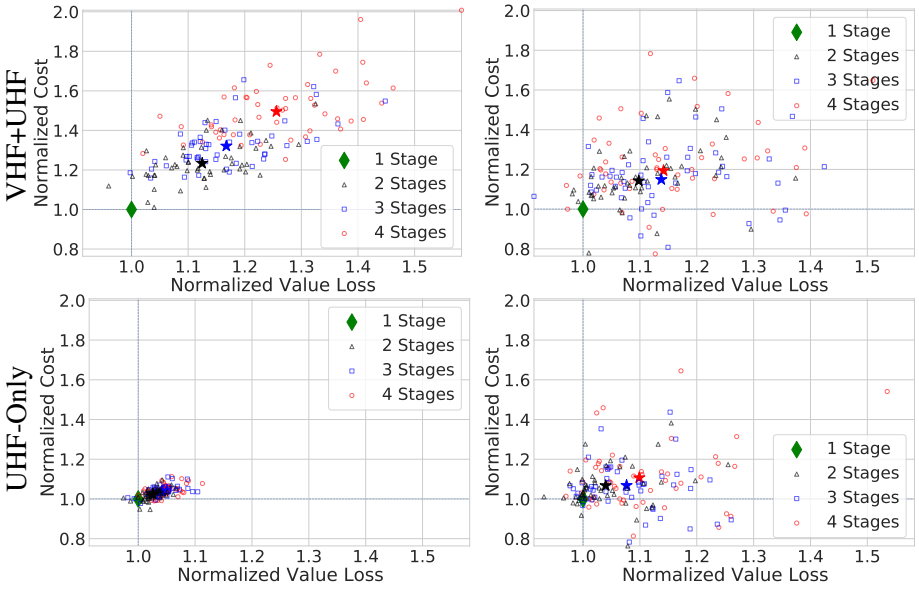
A.5 Full Experimental Results

In this appendix, we provide figures showing the results of our simulations in each of our experiments. Results are shown for both value models and for both auctions that repack and do not repack the VHF band as available.



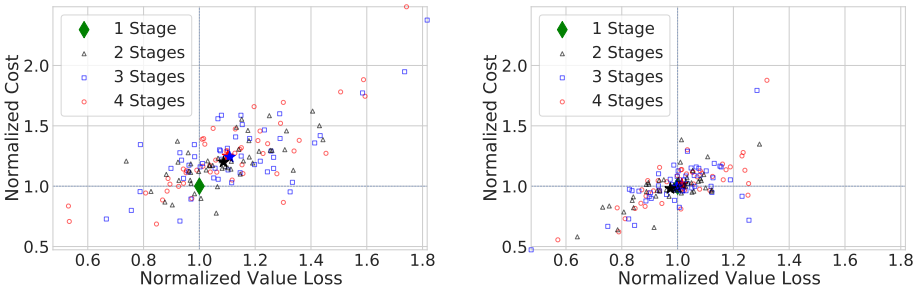
(a) MCS **(b)** BD

Figure A.2: Comparing auctions that only repack the UHF band against auctions that also repack VHF bands.



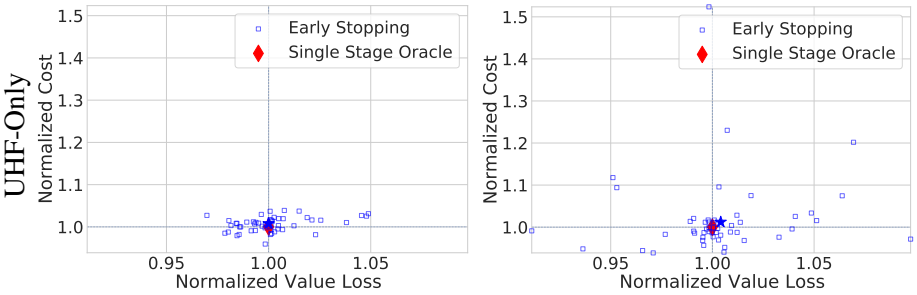
(a) MCS **(b)** BD

Figure A.3: Comparing auctions running through 1-4 stages, ultimately ending on the same clearing target, with no impairing stations.



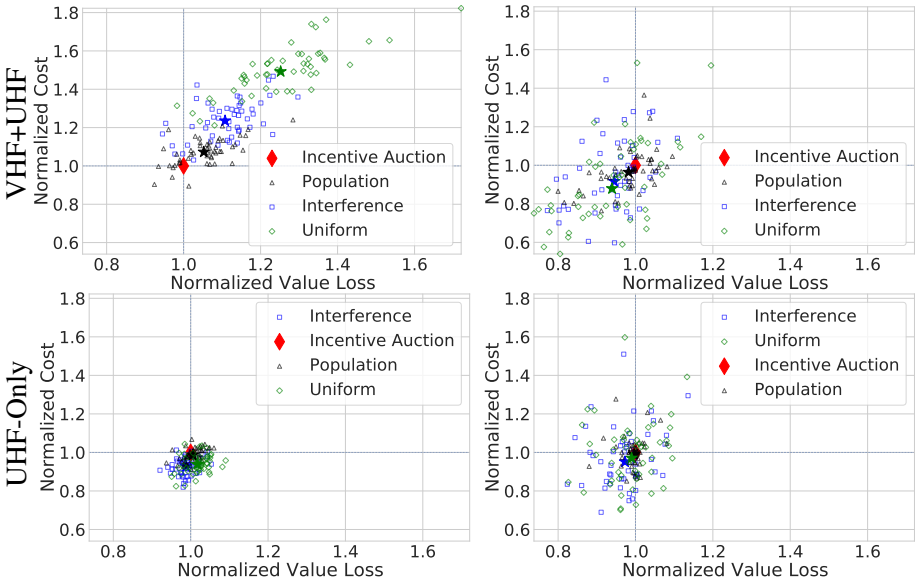
(a) VHF+UHF **(b)** UHF-Only

Figure A.4: Comparing auctions running through 1-4 stages, ultimately ending on the same clearing target, factoring in impairing stations. The BD value model is used for all simulations.



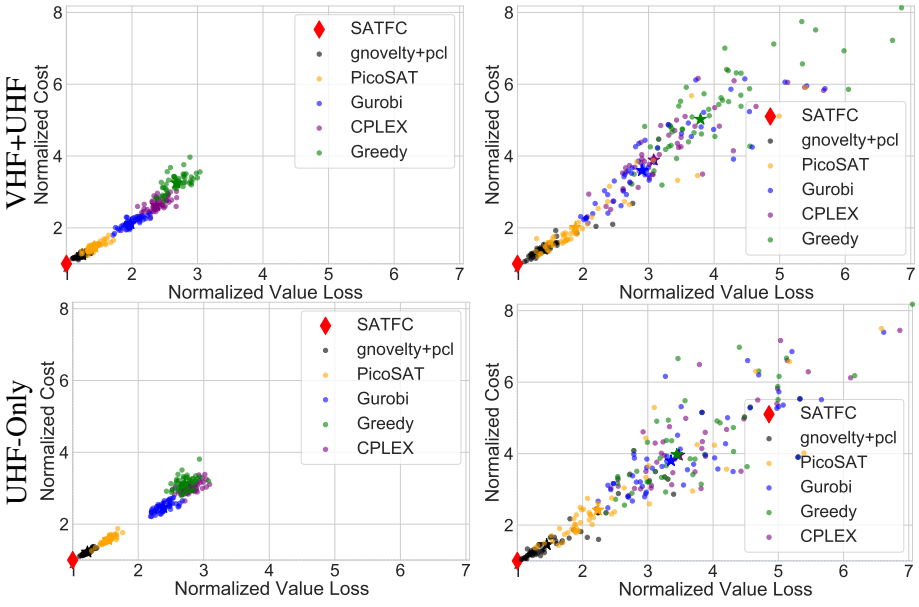
(a) MCS **(b)** BD

Figure A.5: Comparing auctions using the early stopping algorithm against single-stage auctions ending on the same clearing target as their corresponding early stopping auction.



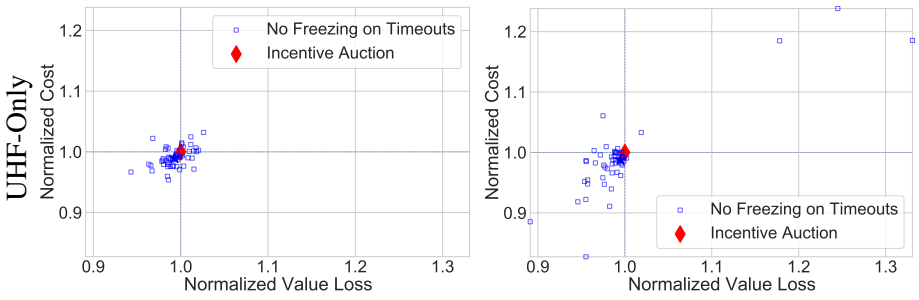
(a) MCS **(b)** BD

Figure A.6: Comparing auctions using four different scoring rules.



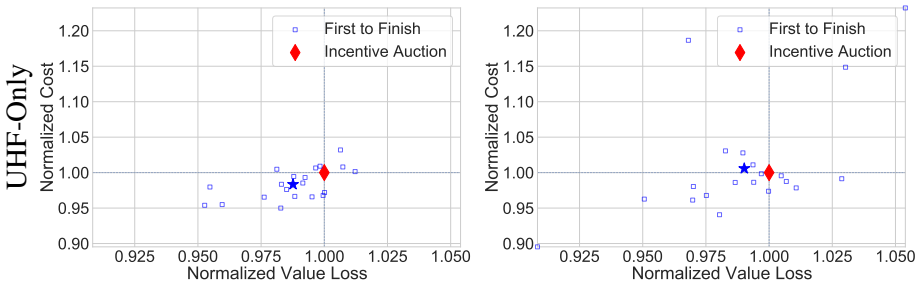
(a) MCS **(b)** BD

Figure A.7: Comparing auctions using different feasibility checkers.



(a) MCS **(b)** BD

Figure A.8: Comparing the standard bid processing algorithm with one that does not freeze stations with indeterminate feasibility checks.



(a) MCS **(b)** BD

Figure A.9: Comparing the first to finish algorithm against the standard bid processing algorithm for single-stage 126 MHz auctions.

A.6 First-to-Finish Algorithm Pseudocode

Algorithm 4 First-to-Finish Bid Processing

Unprocessed $\leftarrow S_{\text{bidding}}$
while Time remains in the round and $|\text{Unprocessed}| > 0$ **do**
    Launch a parallel feasibility check for each $s \in$ Unprocessed, with a cutoff equal to time remaining in the round
    repeat
        Wait for a check to complete
        if s is not frozen **then**
            Process s's bid and remove s from Unprocessed
    until A station exits or moves bands
    Interrupt all ongoing checks

A.7 Impairing Stations

In this appendix, we describe how we choose which set of stations will be impairing given that we cannot replicate the initial clearing target optimization.

Given a clearing target, our simulator determines which stations to impair as follows. First, we ensure full participation of UHF stations by increasing the FCC's base clock price in 5% increments until no UHF stations reject their initial offers. We then solve an optimization problem to find an initial assignment γ that minimizes the number of essential impairing stations, breaking ties by minimizing the aggregate population of impairing stations. Having dealt with the essential impairments, we then run the auction starting from the inflated base clock price. While the base clock price remains higher than its normal starting point, the VHF band is

considered "locked"—only UHF stations are asked to bid and stations cannot bid to move into the VHF band. When the FCC's starting base clock price is reached, any station that would not have participated at the opening prices will have exited or be frozen. We consider any stations that have frozen at this point to be impairing; we do not include impairing stations in any of our metrics unless explicitly noted. At this point, VHF stations make their participation decisions, the VHF band is "unlocked", and the auction proceeds as normal.

In a multi-stage auction, impairing stations may be able to leverage the newly available spectrum to become non-impairing. At the start of each new stage, we again solve an optimization procedure to move as many essential impairments to $\mathcal{C}$ as possible. We then proceed with the between-rounds transition as described in Section 3.2.2, except that p_t can be reset back to the impairment regime, following the same rules as above until the FCC's base clock price is reached. Lastly, we note that there are no essential impairments given a clearing target of 84 MHz and that there are 9 at a clearing target of 128 MHz.

A.8 Robustness Experiments

Recall from Section 4.2.1 that we model a UHF station's value for switching to the HVHF band as $\frac{2}{3} \cdot v_{s,\text{UHF}} \cdot \mathcal{N}(1, 0.05)$ and similarly for the LVHF band with $\frac{1}{3}$ instead of $\frac{2}{3}$. These fractions were chosen for simplicity, drawing on some degree of domain knowledge about the values of VHF bidders. After performing our analysis, we wanted to understand how much our results depended on the specifics of our modelling choices. What if we had used other fractions instead? Here we describe the results of rerunning several experiments under three alternate sets of fractions. The FCC set opening off-air prices for HVHF and LVHF stations at $\frac{3}{5}$ and $\frac{1}{4}$ respectively of corresponding UHF stations. We reran experiments using $\frac{3}{5}$ and $\frac{1}{4}$ as the fractions in our value model. We refer to this parameterization as "Lower Values", as stations have lower values for the VHF bands when compared to our standard value model. For symmetry, we also reran experiments incrementing the fractions by corresponding amounts, leading to $\frac{11}{15}$ and $\frac{5}{12}$. We refer to this parameterization as "Higher Values". We also were interested in whether the Gaussian noise was consequential, so we created another parameterization where

no noise was applied which we refer to as "No Gaussian Noise".

For each of our three value model parameterizations, we reran one experiment related to each question we posed in the introduction. Specifically, we compared: auctions that repacked the VHF band with those that did not ("Repacking VHF"); auctions that ran for four stages against those that simply ran for a single stage beginning at the fourth clearing target ("Clearing Procedure"); auctions that used the FCC's scoring rule against those that only used the population component ("Scoring Rules"); and lastly, auctions that used SATFC 2.3.1 as their feasibility checker against those that used the best off-the-shelf feasibility checker ("Feasibility Checker"). We ran 50 paired simulations for each experiment, just as we did for the corresponding main results. These experiments took roughly 20 CPU years to run.

The results are summarized in Table A.1. We observed slightly different results among the three parameterizations: for example, the cost savings estimate from repacking VHF bands varied from 14–26% depending on how substitutable stations felt the VHF bands were for the UHF band. However, no change to the value model substantially altered the conclusions we drew from any experiment, giving us confidence that our results are relatively robust to the particular choices we made.

A.9 Single-stage vs Multi-stage Auctions Examples

In this appendix, we give an example of how running a single stage auction can lead to a better outcome than iterating through multiple stages (for a fixed final clearing target).

Example A.9.1 *We return to the example in Figure 3.2 and ask what would have happened if the auction had initially set $\overline{c} = 3$ and tried to repack stations into two channels from the outset. We visualize this scenario in Figure A.10. As before, A is the first to exit. However, B does not freeze this time around, because A and B are both repackable using two channels. When $P_t < V_B$, B will exit. At this point C freezes at $\mathcal{P}(C) = V_B$. D is the next to exit, followed by E, finally freezing F at $\mathcal{P}(F) = V_E$. Recall that the value loss and cost of the two stage auction were $V_B + V_F$ and $V_A + V_E$ respectively. In the single-stage alternative, the value loss*

Model Change	Value Loss Repacking VHF	Clearing Procedure	Scoring Rules	Feasibility Checker
Default	0.92 (0.15)	0.95 (0.24)	0.94 (0.08)	1.44 (0.31)
Lower Values	0.96 (0.14)	0.95 (0.18)	0.97 (0.08)	1.51 (0.41)
Higher Values	0.87 (0.15)	0.91 (0.17)	0.93 (0.09)	1.53 (0.33)
No Gaussian Noise	0.9 (0.13)	0.94 (0.19)	0.95 (0.08)	1.48 (0.31)

Model Change	Cost Repacking VHF	Clearing Procedure	Scoring Rules	Feasibility Checker
Default	0.78 (0.15)	0.85 (0.2)	0.96 (0.19)	1.44 (0.29)
Lower Values	0.86 (0.13)	0.89 (0.21)	0.98 (0.18)	1.48 (0.32)
Higher Values	0.74 (0.16)	0.81 (0.17)	0.93 (0.18)	1.55 (0.34)
No Gaussian Noise	0.79 (0.14)	0.85 (0.18)	0.95 (0.19)	1.49 (0.31)

Table A.1: A summary of the results of changing the value model on four replicated experiments from the main paper. Each row corresponds to a different change to the value model, and each column corresponds to an experiment. The "Default" row refers to the original experiment. Values in each cell represent the mean and standard deviation of value loss (upper table) and cost (lower table) of an altered auction design relative to the real auction design across all paired simulations.

is $V_C + V_F$ and the cost is $V_B + V_E$: the outcome is preferable according to both of our metrics!

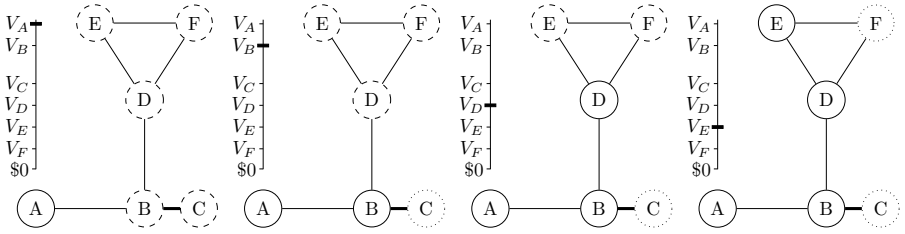


Figure A.10: Returning to the example of Figure 3.2, but now repacking two channels right from the outset.

Next, we provide an example showing that this is not always the case. Here, we show an example where a single stage auction leads to a worse outcome than a multi-stage auction.

Example A.9.2 *Consider a setting with four identically scored stations A, B, C, D with $V_A > V_B > V_C > V_D$, $V_B < V_C + V_D$ and $V_A < 2V_B$. In the final stage, the feasible sets are $\{A, B\}, \{A, C, D\}$ and all subsets of these sets. As prices drop, A exits first, followed by B, which freezes C and D at prices of V_B. So the single-stage approach gives a value loss of $V_C + V_D$ and a cost of $2V_B$*

In the multi-stage setting, in the first stage assume the feasible sets are $\{B\}, \{A, C, D\}$ and all subsets of these sets. A will exit first, freezing B at a price of V_A. B and C then exit, concluding the stage. In the second stage, B never unfreezes. Therefore, the total cost of the multi-stage approach is V_A and the value loss is V_B. By the inequalities assumed above, this is a cheaper, more efficient outcome than the single-stage auction that predetermined the amount of spectrum to clear.

A.10 Early Stopping Counterexample

In this appendix, we provide an example where early stopping leads to a worse outcome than the standard clearing procedure.

Example A.10.1 *Consider four identically scored stations A, B, C, D with $V_A > V_B > V_C > V_D$, $V_B < V_C + V_D$ and $V_A < 2V_B$. Let the forward auction run first and have a purchasing price of V_A. Let the feasible sets in the first stage be $\{A, C\}, \{A, D\}, \{B\}$ and all subsets of these sets. In the second stage, the feasible sets add $\{A, B\}, \{A, C, D\}$ and all subsets. In both cases, the auction begins with A exiting and B freezing at price V_A. In an early stopping auction, this will trigger the end of stage one. In stage two, B unfreezes and exits. Then C and D freeze at price V_B. This leads to a value loss of $V_C + V_D$ and a cost of $2V_B$. In an auction without early stopping, C would exit, freezing D at price V_C. This would trigger the stage to end. In the next stage, B would remain frozen and D would unfreeze and exit, leading to a value loss of V_B and a cost of V_A. Using the inequalities on the values above, the auction that does not use early stopping performs better in both metrics.*

A.11 Feasibility Checker Counterexample

In this appendix, we provide an example showing that a better feasibility checker does not necessarily lead to better outcomes, and can even lead to worse ones.

For a fixed cutoff, a feasibility checker can be thought of as a mapping from a set of stations to $\{Feasible, Infeasible, Unknown\}$. We can define an ordering over feasibility checkers such that a feasibility checker F_1 is strictly better than a second F_2 if and only if for all possible sets of stations $s \in 2^S$, $F_2(s) = Feasible \implies F_1(s) = Feasible$ and $\exists s$ such that $F_1(s) = Feasible$ and $F_2(s) = Unknown$. Importantly note that for the purposes of this definition we don't care about the feasibility checker's ability to prove infeasibility: while this is important for saving time, it does not ultimately impact the result of the auction since infeasibility and indeterminate solutions are treated identically. We now proceed to the example.

Example A.11.1 *Imagine a UHF-only setting involving four bidding stations A, B, C, and D. Let $V_A = V_B = V_C = V_D = V$ and let all stations have the same score. The constraints are such that the repackable sets are either $\{B, C, D\}$ or $\{A, D\}$ (and all subsets). There are two feasibility checkers: F_1 can find all feasible repackings, but $F_2(\{A, D\}) =$ Unknown. Consider the first round in which each station is being offered a price p just below V and assume that the bid processing order is D, A, B, C. D exits the auction. Under F_1, A is allowed to exit the auction. This freezes B and C. The value loss for these two stations will be $2V$ and the payment will be just under $2V$. F_2, however, cannot pack A, and so A freezes. B and C then exit the auction. The value loss in this scenario is V and the payment is just under V, so this outcome is strictly better than the previous even though F_1 is strictly better than F_2.*

A.12 Modeling Forward Auction Revenue

When using the early stopping algorithm, the reverse auction takes as input the output of the previous forward auction (the amount that mobile carriers will pay for the spectrum). Therefore, in order to simulate early stopping auctions, we need to model forward auction revenues.

While we have access to the real forward auction revenues, the incentive auction only went through four stages, and simulations could potentially go to stages beyond the fourth. To address this issue, we performed a log-log fit on the number of mobile licenses and forward auction proceeds in the first three stages (i.e., with 3 data points we fit $ln(\#\text{licenses}) = a \cdot ln(cost) + b$ for some constants a, b). We ignored the revenue in the fourth stage when performing this fit because the price per license rose significantly relative to the other stages, and we suspect that the price increase was likely due to an understanding among bidders that the auction would terminate in this stage. After the fit was established, we scaled the entire model by a constant so that the model's prediction for the fourth stage matched the observed real revenue. The results appear in Figure A.11.

We used the same forward auction revenues across all sampled station value profiles for a given value model. When using the MCS model, we observed that reverse auction simulations rarely produced high enough procurement costs to trigger early stopping in the first stage, leading to uninformative behavior in which the auction simply ends after the first stage. To get around this, we scaled our forward auction model downwards by a factor of 2 when running simulations for the MCS.

A.13 VCG MIP Encoding

We note that stations are divided into two sets: participating stations $S_{\text{participating}}$ and non-participating stations $S_{\text{non-participating}}$. Importantly, if a station does not participate, it must remain on air, whereas bidding stations need not be assigned a channel. We create a Boolean variable $x_{s,c} \in \{0, 1\}$ for every station–channel pair $(s, c) \in S \times C$, representing the proposition that station s is assigned to channel c. Let $D(s)$ be a function that returns the domain (set of available channels) for s, and I be the set of all invalid pairs of station-channel assignments (s, c, s', c'). We encode the packing problem as a MIP with the objective of maximizing the value

of on-air stations.

$$\text{maximize} \sum_{s \in S_{\text{participating}}} \sum_{c \in D(s)} x_{s,c} \cdot v_{s,\text{pre}(s)}$$

$$\text{subject to } x_{s,c} + x_{s',c'} \leq 1 \qquad \forall \{(s,c), (s',c')\} \in I \qquad \text{(A.9)}$$

$$\sum_{c \in D(s)} x_{s,c} \leq 1 \qquad \forall s \in S \qquad \text{(A.10)}$$

$$\sum_{c \in D(s)} x_{s,c} = 1 \qquad \forall s \in S_{\text{non-participating}} \qquad \text{(A.11)}$$

$$x_{s,c} \in \{0,1\} \qquad \forall c \in D(s) \, \forall \, s \in S \qquad \text{(A.12)}$$

The objective function expresses that we want to maximize the sum of values of participating stations that remain on air. Constraint (A.9) prohibits infeasible channel assignments, Constraint (A.10) says that every station is either packed on a single channel or else is not packed, and Constraint (A.11) says that we must find a channel for all non-participating stations. Lastly, Constraint (A.12) requires that each station's channel indicator variables be integral. Solving the MIP will yield the efficient allocation.

A.14 Station Repacking Benchmarks

The exact repacking problems that arose in the incentive auction were not made publicly available. We believe that station repacking problems represent an important industrial problem benchmark, and that training examples of these problems may prove useful to the broader empirical algorithms community (indeed, part of this benchmark was submitted to the 2020 SAT Competition Balyo et al. [2020] without our knowledge or consent!). Hence, in addition to releasing our simulator (allowing users to generate their own repacking problems), we also released a set of benchmarks based on our experiments in Chapter 4 to share with the community. The full set of instances are available as CNFs at http://cs.ubc.ca/labs/beta/Projects/SATFC. The previous solution γ^- is also available for each problem.

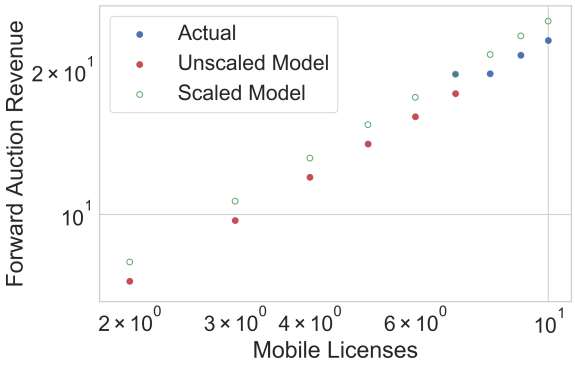


Figure A.11: Forward auction revenues observed in the incentive auction ("Actual") and our model of forward auction revenue.

Appendix B

Supporting Materials: Reinforcement Learning

B.1 Computational Environment

Our experiments were performed on two different compute clusters. Our first cluster consisted of nodes equipped with 32 2.10GHz Intel Xeon E5-2683 v4 CPUs with 40960 KB cache and 96 GB RAM. Our second cluster consisted of 16-core machines with Intel Silver 4216 Cascade Lake 2.1GHz processors and 96 GB RAM. Jobs were scheduled such that each run had access to 4 CPUs and 20 GB of RAM.

B.2 Clock Auctions

In this appendix, we give a more formal description of the clock auction games that we implemented.

Let N be the number of bidders and M be the number of service areas. Each service area j contains q_j identical licenses for sale.[1] The clock auction proceeds over a series of rounds t, beginning with $t = 1$. In each round t, all bidders observe a *start-of-round price* $P_{j,t}$ and a *clock price* $P_{j,t} \cdot a$ for each product j,

[1] A realistic auction, e.g. Canada's 600 MHz auction had $N = 10$ and $M = 15$. Canada's 3500 MHz auction had M much higher, in the hundreds). Typical values for q_j range from 1 to 10.

where a is the *clock increment* (e.g., 5–20%). (These prices are anonymous—i.e., all bidders face the same price.) Then, each bidder i simultaneously submits a bid $B_{i,t,j} \in \{0, 1, \ldots, q_j\}$ for each product j, representing their demand for each product. After the bids are made, each product has an aggregate demand $Z_{j,t} = \sum_i B_{i,j,t}$. If no product is overdemanded (i.e., $Z_{j,t} \leq q_j$ for all products j), then the auction ends, with each bidder i winning $B_{i,t,j}$ licenses of product j, paying $P_{t,j}$ for each one. Otherwise, the price of each overdemanded product increases, with the start-of-round price being set to the previous clock price, and the auction continues on to the next round.

Typically, auctions include an *activity* rule. The rule restricts the set of legal bids based on a bidder's bid history with the aim of discouraging strategic bidding. While there are many variants, a simple and common rule is to assign each product j a number of *eligibility points* e_j; then, each bidder's total activity $\sum_j B_{i,t,j} \cdot e_j$ must be non-increasing. This rule disallows bidders from bidding for few items early in the auction and rapidly expanding their bid later.

Auctions can allow for intra-round bidding. Such a provision allows bidders to express changes in their demand between clock increments (e.g., if the start-of-round price is \$100 and the clock price is \$120 and a bidder holds two licenses in a region, they may bid to drop to one license at \$110. If this drop causes supply to equal demand, the new start-of-round price would be \$110.). Intra-round bidding allows an auctioneer to set larger clock increments. Perhaps our most significant modeling simplification is that we do not model intra-round bidding. Instead, we assume that agents can *only* bid at the clock price. This restriction requires the clock increments in our modeled games to be kept relatively small.

The clock auction includes a rule that prohibits demand from dropping ever dropping from above supply to below supply. This guarantees that once a product has ever had a round where demand was at least supply, it will be sold. In practice, this requires the auctioneer to reject some bids. The bid processing algorithm works by placing tuples of *requests* of the form (bidder, product, change in demand) into a queue. These requests are then processed in queue order to the degree possible (a request may only be partially fulfilled if either there is not enough demand to allow further drop bids, or the bidder does not have sufficient activity to allow a pick up bid). If a request is only partially fulfilled, it is reinserted into the

queue. The algorithm continues applying these requests until it is not able to apply any additional requests in the queue.

B.3 Value Sampling Details

Removing uninteresting games. Our game sampler rejects games that we deem strategically uninteresting. We require that no bidder, in all type realizations, is allocated nothing under the utility-maximizing allocation at opening prices. We solve for the utility-maximizing allocation using a MIP. Such a bidder is hopelessly weak, tending to slow down simulations for no added benefit.

Infeasible games. Some criteria ensured that we generated games that we were capable of solving. First, as a proxy measure for the length of the auction, we computed the number of rounds it would take for the auction to conclude in a simulation in which each bidder bids for its profit-maximizing bundle at start-of-round prices in every round. We rejected auctions longer than 20 such rounds. Secondly, we reran the above simulation, this time incrementing the price on a single, randomly chosen product among those that were overdemanded in each round. This is a crude model of the worst-case auction (from a length perspective), in which each round prices only increment on a single product. We rejected simulations that took on average longer than 25 such rounds to conclude. Lastly for 2-player games, we required MCCFR to be able to run at least 25 iterations per second. These checks were all based on the drop-by-player rules.

Changes for 3-player games. We made several small changes to the value model for the 3-player games in Section 6.4.3. For MRVM parameters, bidders had market shares sampled uniformly from 20% to 30% and values per subscriber sampled uniformly from \$35 million to \$45 million. The keypoints on their sigmoidal value function were placed at $\pm 10\%$ from their market share. We also adjusted the rejection sampler requirement that MCCFR be able to run 25 iterations per second, loosening this threshold to 1 iteration per second.

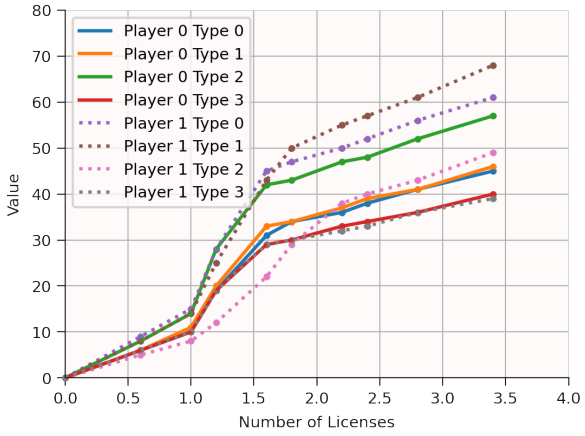


Figure B.1: An example value profile for a game with two bidders and four types. Fractional numbers of licenses on the x-axis correspond to bundles with encumbered licenses.

Example value function. We show an sample value function in Figure B.1.

B.4 PPO Hyperparameters

Our configurations were sampled as follows:

```
def sample_config():
    config = dict()

config['steps_per_batch'] = int(np.random.choice([64, 128]))
config['num_minibatches'] = int(np.random.choice([4, 8]))
config['update_epochs'] = int(np.random.choice([4, 8, 16]))
config['learning_rate'] = float(np.random.choice([3e-5, 6e-6, 9e-5, 1e-4, 3e-4]))
config['gae'] = bool(np.random.choice([True, False]))
config['anneal_lr'] = bool(np.random.choice([True, False]))
config['gae_lambda'] = float(np.random.uniform(0.94, 1.))
config['clip_coef'] = float(loguniform.rvs(0.0003, .3, size=1)[0])
config['clip_vloss'] = bool(np.random.choice([True, False]))
config['entropy_coef'] = float(np.random.choice([1e-4, 1e-5, 1e-6, 0, 3e-5, 3e-6]))
config['value_coef'] = float(loguniform.rvs(0.1, 1.3, size=1)[0])
config['num_envs'] = int(np.random.choice([4, 8, 16]))
config['normalize_advantages'] = bool(np.random.choice([True, False]))
```

```
config['optimizer'] = random.choice(['adam', 'rmsprop', 'sgd'])
optimizer_kwargs = dict()
if config['optimizer'] == 'adam':
    beta1 = float(np.random.uniform(0.8, .9))
    beta2 = float(np.random.uniform(0.8, .999))
    optimizer_kwargs['betas'] = [beta1, beta2]

config['optimizer_kwargs'] = optimizer_kwargs
config['max_grad_norm'] = float(np.random.uniform(0.1, 1))

agent_kwargs = dict()
agent_fn = base_config['agent_fn'].lower()
if agent_fn == 'auctionnet':
    agent_kwargs['activation'] = str(np.random.choice(['relu', 'tanh']))
    agent_kwargs['hidden_sizes'] = random.choice([
        [32, 32, 32], [64, 64, 64], [128, 128, 128],
    ])
    agent_kwargs['add_skip_connections'] = bool(np.random.choice([True, False]))
    agent_kwargs['use_torso'] = bool(np.random.choice([True, False]))
elif agent_fn == 'ppoagent': # MLP
    hidden_sizes = random.choice([
        [64, 64], [128, 128], [256, 256],
        [64, 64, 64], [128, 128, 128], [256, 256, 256],
    ])
    activation = str(np.random.choice(['relu', 'tanh']))
    agent_kwargs['actor_hidden_sizes'] = hidden_sizes
    agent_kwargs['critic_hidden_sizes'] = hidden_sizes
    agent_kwargs['actor_activation'] = activation
    agent_kwargs['critic_activation'] = activation

config['agent_fn_kwargs'] = agent_kwargs

return config
```

Our final PPO configuration is as follows:

	Variable	Value
0	Activation	relu
1	Add Skip Connections	True
2	Anneal LR	True
3	Architecture	AuctionNet
4	Clip Coef	0.018894
5	Clip Vloss	False
6	Entropy Coef	0.000030
7	Gae	False
8	Gae Lambda	0.985872
9	Gamma	1.000000
10	Hidden Sizes	[32, 32, 32]
11	Learning Rate	0.000090
12	Max Grad Norm	0.526474
13	Normalize Advantages	False
14	Num Annealing Updates	—
15	Num Envs	16
16	Num Minibatches	8
17	Optimizer	adam
18	Optimizer Betas	[0.8032349345217956, 0.8653019218632017]
19	Steps Per Batch	64
20	Target KL	—
21	Track Stats	True
22	Update Epochs	8
23	Use Torso	True
24	Value Coef	0.609288

B.5 Additional Results

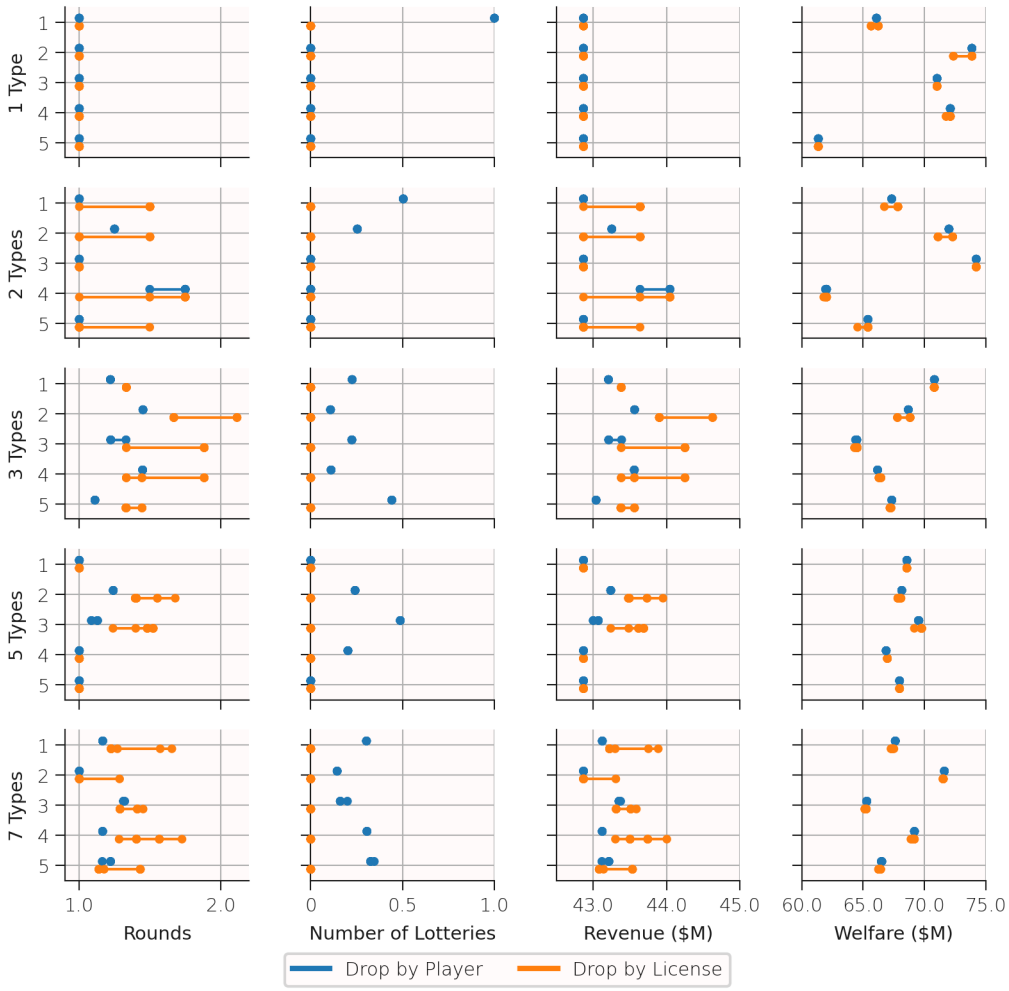


Figure B.2: Auction outcomes using MCCFR on 2-player games with 1 to 7 types.

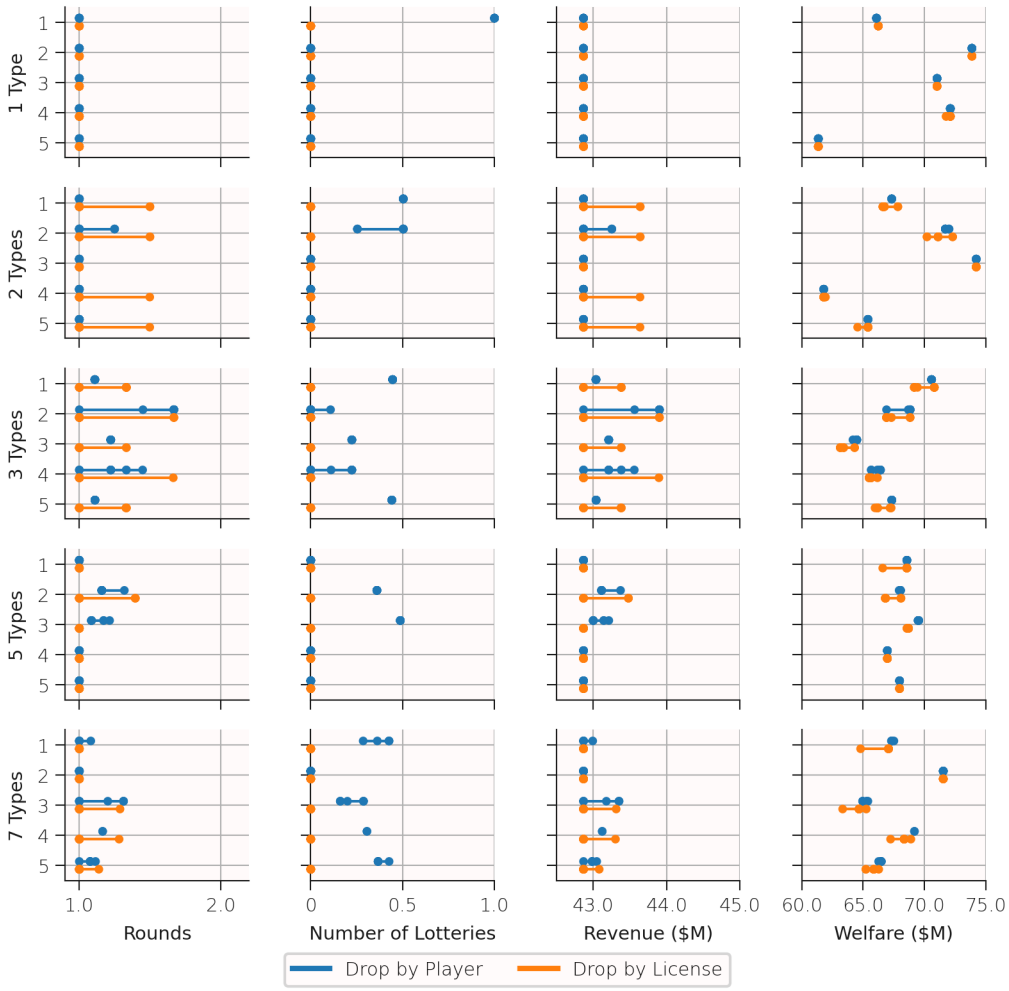


Figure B.3: Auction outcomes using PPO on 2-player games with 1 to 7 types.

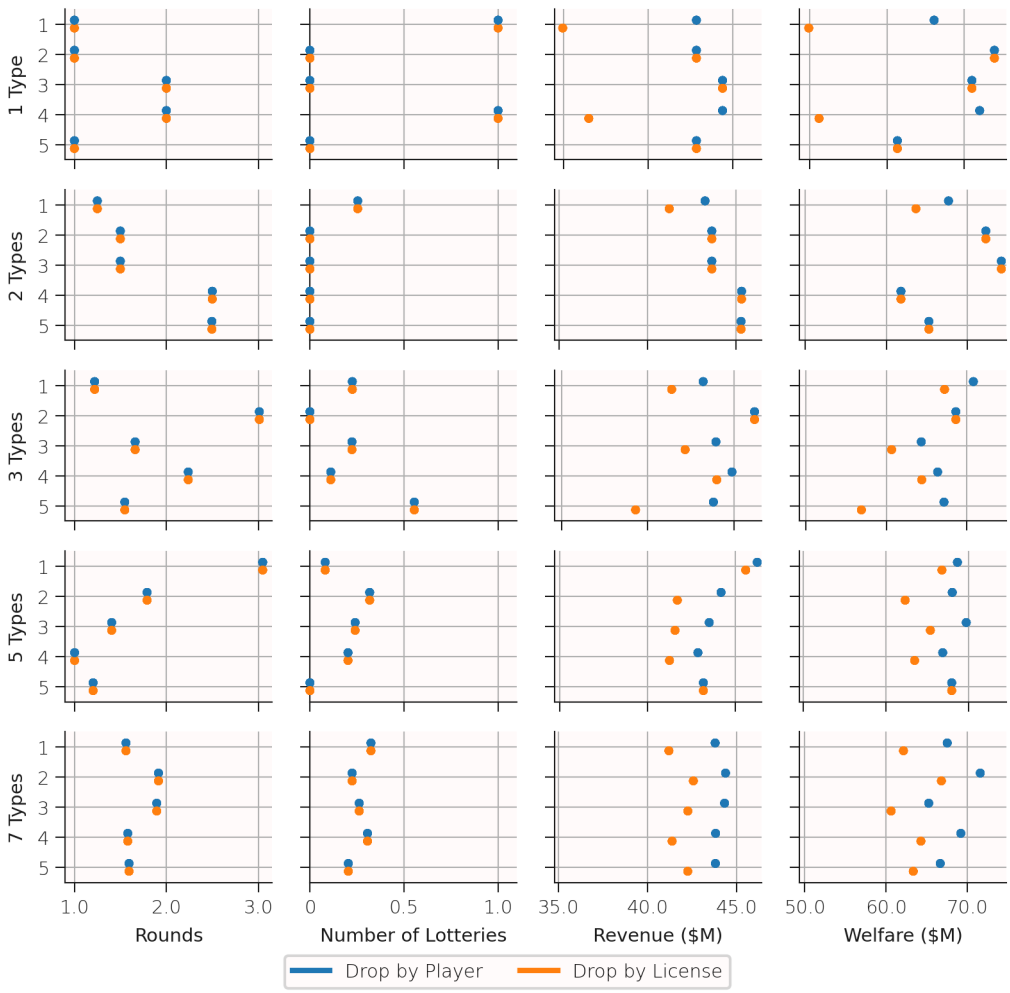


Figure B.4: Auction outcomes under straightforward bidding on 2-player games with 1 to 7 types.