



Issues in Scaling Up the 700 MHz Auction Design

*Wye River Conference II
October 27, 2001*

Melissa Dunford
Karla Hoffman
Rudy Sultana

Martin Durbin
Dinesh Menon
Thomas Wilson



Introduction

- ▲ Auction #31 rules were designed specifically for that auction
 - No computational problems for a 12 license auction
 - Known scalability issues
- ▲ Seeking a design for large combinatorial auctions
- ▲ Start with Auction #31, identify the problem areas in regards to scalability and then discuss alternatives



Outline

- ▲ Review of Auction #31 rules impacting scalability and possible alternatives
 - Determining maximum revenue each round
 - Choosing among ties
 - Minimum Acceptable Bid (MAB)
- ▲ Mechanism for testing scalability -- BidBots
- ▲ Test design
- ▲ Ideas on where to go from here



Rules of Auction #31

Impacting Scalability

- ▲ Solve for the maximum revenue *exactly* each round
 - Considered bids for each bidder in a round consist of:
 - ▲ Bids (new and renewed) made by the bidder in the last two rounds in which the bidder placed bids
 - ▲ The bidder's provisionally winning bids from the previous round
- ▲ *Randomly* choose among ties
- ▲ Part *iii* of the minimum acceptable bid rule
 - A bidder's previous high bid on a license/package *plus a share of the increase in revenue needed to tie the provisional winners*
- ▲ Activity Calculations



Auction #31 Formulation

- ▲ Choose among a set of bids such that:
 - Revenue to the FCC is maximized
 - Each license is awarded exactly once
 - No bidder has bids in a provisionally winning bid set from more than one round

$$\underset{x}{Max} \sum_{b=1}^{\#Bids} BidAmt_b x_b$$

subject to:

$$Ax = 1 \quad (\text{each license awarded exactly once})$$

Mutually Exclusive Bid Constraints

$$x_b \in \{0,1\} \quad \text{for all bids}$$



$Ax = 1$: Each license awarded once

Example: 4 licenses, 8 bids

Bid	1	2	3	4	5	6	7	8
Bid amt.	\$22e6	\$12e6	\$30e6	\$16e6	\$8e6	\$11e6	\$10e6	\$7e6
Package	ABD	B	ABC	AD	C	BC	A	D

A	1	0	1	1	0	0	1	0	=	1
B	1	1	1	0	0	1	0	0	=	1
C	0	0	1	0	1	1	0	0	=	1
D	1	0	0	1	0	0	0	1	=	1

- ▲ This is called a *set-partitioning* problem. These types of problems have a very nice mathematical structure.



Mutually Exclusive Bid Rule

- ▲ Bids made by the same bidder in different rounds are treated as *mutually exclusive* (“or”) bids.
 - Thus for each bidder, any bids in the provisionally winning set must have been made in the same round.
 - If a bidder does not want a bid from a previous round (including a provisionally winning bid) to be considered mutually exclusive with bids made in the current round, it must renew or increase the bid in the current round.
- ▲ Implementing this rule requires that new, “dummy” variables be created
 - For each round in which a bidder placed bids that are being considered by the solver, create a “**use-round**” variable. The variable will equal 1 if the round is “used” and 0 otherwise.



Example: Mutually Exclusive Bid Constraints

Considered Bids for Bidder A

Set of bids made in Round 1: $X_{(A,1)} = \{x_1, x_2, x_3, x_4, x_5\}$

Set of bids made in Round 2: $X_{(A,2)} = \{x_6, x_7, x_8\}$

Create two “use-round” variables for Bidder A: $u_{(A,1)}, u_{(A,2)} \in \{0,1\}$

Form the following constraints:

$$x_1 + x_2 + x_3 + x_4 + x_5 \leq 5u_{(A,1)}$$

$$x_6 + x_7 + x_8 \leq 3u_{(A,2)}$$

$$u_{(A,1)} + u_{(A,2)} \leq 1$$



Solving Challenges: Mutually Exclusive Bid Rule

- ▲ Mutually exclusive bid constraints alter the nice structure of set-partitioning ($Ax = 1$) and make the problem harder to solve

A	1	0	1	1	0	0	1	0	1	0	=	1
B	1	1	1	0	0	1	0	0	0	0	=	1
C	0	0	1	0	1	1	0	0	0	0	=	1
D	1	0	0	1	0	0	0	1	0	1	=	1
R_(A,1)	1	1	1	1	1	0	0	0	-5	0	<=	0
R_(A,2)	0	0	0	0	0	1	1	1	0	-3	<=	0
U_A	0	0	0	0	0	0	0	0	1	1	<=	1



Solving Challenges: Mutually Exclusive Bid Rule

- ▲ Mutually exclusive bid constraints alter the nice structure of set-partitioning ($Ax = 1$) and make the problem harder to solve
- ▲ Mutually exclusive bid constraints force all bids to be considered
 - Identical bids with different bid prices cannot be removed before solving
- ▲ For *each bidder* with considered bids, 3 or 4 extra constraints must be added to the problem



Mutually Exclusive Bid Rule

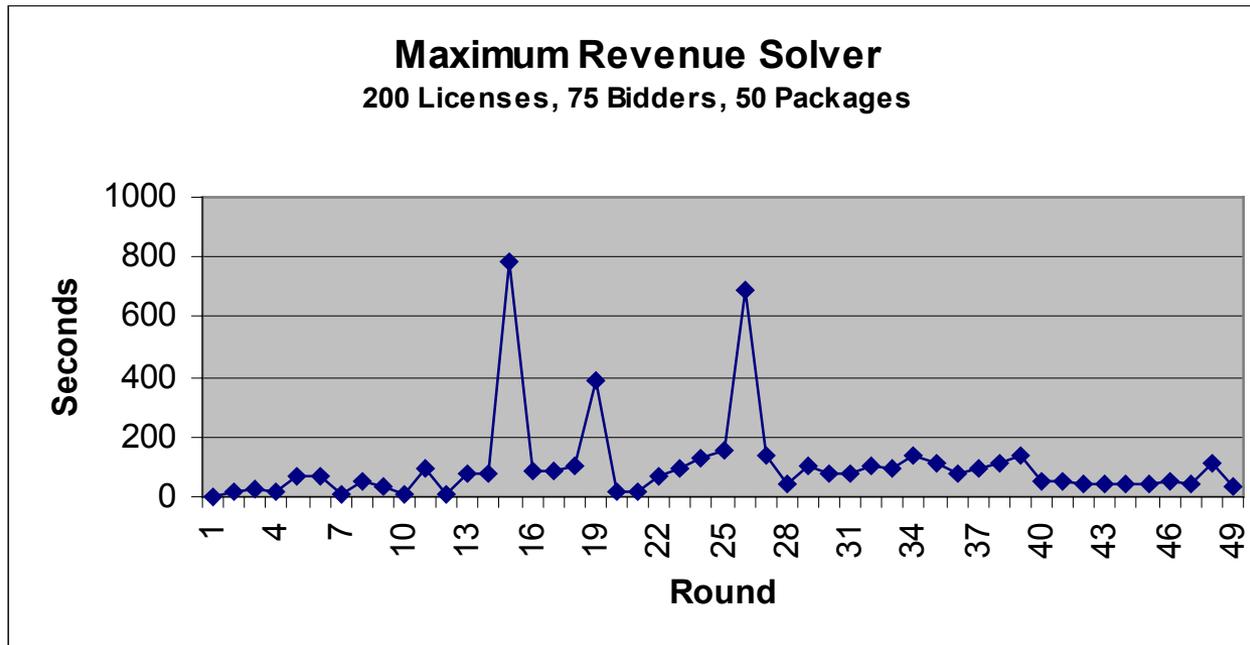
- ▲ Why use a mutually exclusive bid rule?
 - Used to reduce the risk of a bidder winning more than it wants
 - ▲ e.g. budget limitations, alternative business plans
 - Used to assure that bids remain for at least two rounds to overcome threshold problem

- ▲ Alternative mutually exclusive bid rules
 - Pseudo licenses (Fujishima, Leyton-Brown, Shoham)
 - *All bids* of a bidder are mutually exclusive with each other (Ausubel, Milgrom)
 - Bidder wants no more than k out of n bids to win
 - Budget constraint



Solvability of Integer Optimizations

- ▲ The size and complexity of “solvable” integer problems is growing at an extraordinary rate
- ▲ In our testing most problems solve very quickly, but...





Tie-Breaking Procedure

- ▲ Ties will be broken randomly using a two-step process
 - First step uses optimization to select a bid set that achieves the maximum revenue
 - Second step uses optimization to guarantee that the bid set found in step one is unique
 - ▲ To date a unique set has always been found in the first step

*Note: Breaking ties randomly means that a package that was a provisionally winning bid in round k and is in some tied provisionally winning set in round $k+1$, might **not** be chosen as a winning bid in round $k+1$*



Choosing Randomly Among Winning Sets

- ▲ Each considered bid is assigned a *selection number*
 - A bid's selection number is the sum of n pseudo-random numbers where n is the number of licenses comprising the bid's package

$$\text{Max}_x \sum_{b=1}^{\#Bids} \text{SelectNum}_b x_b$$

subject to:

$$Ax = 1 \quad (\text{each license awarded exactly once})$$

Mutually Exclusive Bid Constraints

$$\sum_{b=1}^{\#Bids} \text{BidAmt}_b x_b = \text{Maximum Revenue}$$

$$x_b \in \{0,1\} \quad \text{for all bids}$$



Solving Challenges: Tie-Breaking

- ▲ The optimization required to break ties is a harder problem to solve than the maximum revenue problem due to a machine arithmetic problem

A	1	0	1	1	0	0	1	0	1	0	= 1
B	1	1	1	0	0	1	0	0	0	0	= 1
C	0	0	1	0	1	1	0	0	0	0	= 1
D	1	0	0	1	0	0	0	1	0	1	= 1
$R_{(A,1)}$	1	1	1	1	1	0	0	0	-5	0	≤ 0
$R_{(A,2)}$	0	0	0	0	0	1	1	1	0	-3	≤ 0
U_A	0	0	0	0	0	0	0	0	1	1	≤ 1
Rev	<i>22e6</i>	<i>12e6</i>	<i>30e6</i>	<i>16e6</i>	<i>8e6</i>	<i>11e6</i>	<i>10e6</i>	<i>7e6</i>	0	0	= <i>37e6</i>



Choosing Randomly Among Ties

- ▲ Why use this method of breaking ties?
 - Solvers work at getting an optimal solution quickly, no solver automatically provides tie information
 - Breaking ties randomly seems fair
 - Cannot use a tie-breaking method that requires the generation of all tied sets (there could be millions of tied sets even for an auction with few licenses)
- ▲ Current method of random tie-breaking works
- ▲ Due to time considerations in large auctions, tie-breaking procedure may be postponed until later rounds
 - Concept of stages



Minimum Acceptable Bids

A minimum acceptable bid is the *greater* of:

- i. The minimum opening bid
- ii. The bidder's previous high bid on a license/package plus $X\%$
- iii. The bidder's previous high bid on a license/package plus a share of the revenue needed to tie the provisional winners
(Pekec, Rothkopf; Weber; Milgrom)



Shortfall Formulation

- ▲ Determining Shortfall Revenue for Bid i

$$\text{Max}_x \sum_{b=1}^{\#Bids} \text{BidAmt}_b x_b$$

subject to:

$$Ax = 1 \quad (\text{each license awarded exactly once})$$

Mutually Exclusive Bid Constraints

$$x_i = 1$$

$$x_b \in \{0,1\} \quad \text{for all bids}$$

Bid i 's Shortfall = Maximum Revenue - Shortfall Revenue _{i}



Distributing Shortfall

- ▲ Since there can be more than one set of bids that produce bid i 's shortfall revenue, we choose to pick the set with the most provisional winning bidding units and distribute the shortfall among the non-winning bids in the set.

$$\text{Max}_x \sum_{w \in W} \text{BiddingUnits}_w x_w \quad W = \{\text{winning bids}\}$$

subject to:

$$Ax = 1 \quad (\text{each license awarded exactly once})$$

Mutually Exclusive Bid Constraints

$$\sum_{b=1}^{\#Bids} \text{BidAmt}_b x_b = \text{Shortfall Revenue}_i$$

$$x_i = 1$$

$$x_b \in \{0,1\} \quad \text{for all bids}$$



Deficit Calculations

- ▲ Determining a deficit for Bid i

$$NonWinningBU_i = TotalBU - MaxWinningBU_i$$

$$Deficit_i = \frac{BU_i}{NonWinningBU_i} * Shortfall_i$$

$$Part\ 3\ MAB_i = BidAmt_i + Deficit_i$$



Scaling-up Auction #31

- ▲ Largest auction run with 10 min round results limitation
 - 50 licenses, 25 bidders, 25 packages
- ▲ Deficit calculations are the biggest “bottle-neck”
- ▲ Reason:
 - A deficit is calculated for *every* license and *every* constructed package of *every* bidder
 - ▲ Up to $(\#Licenses + \#Packages) * \#Bidders$, calculations per round
 - *Each* deficit calculation requires solving two *integer* programs
- ▲ On average, 99.5% of the runtime was spent calculating deficits



Alternatives to Deficit Calculations

- ▲ Many authors suggest removing rule iii
 - Auction takes too long to close at 10% increment
 - Raising increment percentage may create a threshold problem
- ▲ Other authors suggest using *shadow prices* to calculate MABs.
 - ▲ Rassenti, Smith, Bulfin (1982)
 - ▲ DeMartini, Kwasnisca, Ledyard, Porter (1999)
 - ▲ Milgrom (2001)
 - Shadow prices are the dual prices of the *linear* programming approximation to the integer problem
 - **Shadow prices estimate the current value of each license**
 - License values can be used in determining MABs



Alternative Rule iii

Shadow Prices

A minimum acceptable bid is the *greater* of:

- i. The minimum opening bid
- ii. The bidder's previous high bid on a license/package plus $X\%$
- iii. The estimated value of a license/package plus $Y\%$

Note: The estimated value of a package is the sum of the shadow prices of the licenses that make up the package



Issues With Using Shadow Prices

- ▲ The linear program is an approximation to the integer program. The linear program can overestimate the integer programming solution.
 - Thus, the sum of the shadow prices might be greater than the maximum revenue
- ▲ There may be multiple alternative shadow prices that yield the same revenue. How should we choose among them?
- ▲ Does using shadow prices for setting MABs impact other optimizations?



Auction Formulation

License	A	B	C
---------	---	---	---

Bid	1	2	3	4	5
Bid amt.	\$22	\$24	\$20	\$16	\$8
Package	AB	BC	AC	AB	C

Bid	1	2	3	4	5	Objective Value: \$30	
Maximize	\$22 b_1	+ \$24 b_2	+ \$20 b_3	+ \$16 b_4	+ \$8 b_5		
License A	b_1	+ 0	+ b_3	+ b_4	+ 0	\Leftrightarrow	Count 1
License B	b_1	+ b_2	+ 0	+ b_4	+ 0	\Leftrightarrow	1
License C	0	+ b_2	+ b_3	+ 0	+ b_5	\Leftrightarrow	1
Variables	1	, 0	, 0	, 0	, 1	=	0 or 1

Note: This formulation of the maximum revenue problem ignores the mutual exclusive bid constraints.



Dual Formulation

LP Formulation

Objective Value: \$33

Bid	1	2	3	4	5		Count
Maximize	$\$22 b_1$	$\$24 b_2$	$\$20 b_3$	$\$16 b_4$	$\$8 b_5$		
License A	b_1	0	b_3	b_4	0	\Leftrightarrow	1
License B	b_1	b_2	0	b_4	0	\Leftrightarrow	1
License C	0	b_2	b_3	0	b_5	\Leftrightarrow	1
Variables	0.5	0.5	0.5	0	0	\Downarrow	0

Dual Problem

Objective Value: \$33

License	A	B	C		Bid amt.
Minimize	$1s_A$	$1s_B$	$1s_C$		
Bid 1	s_A	s_B	0	\Downarrow	\$22
Bid 2	0	s_B	s_C	\Downarrow	\$24
Bid 3	s_A	0	s_C	\Downarrow	\$20
Bid 4	s_A	s_B	0	\Downarrow	\$16
Bid 5	0	0	s_C	\Downarrow	\$8
Variables	9	13	11	\Downarrow	0



Solution: Pseudo-Duals

- ▲ Rassenti, Smith and Bulfin (1982) as well as DeMartini, Kwasnica, Ledyard and Porter (1999) suggest solving for pseudo-dual prices in order to adjust the over-estimated shadow prices
- ▲ The idea is to use the provisional winners as the “mark” for adjusting down the shadow prices
- ▲ Result...
 - The shadow prices sum to equal the revenue obtained from the integer solution
 - The sum of the shadow prices of a winning package equals the bid amount of the package



Adjusted Shadow Prices

Pseudo-Dual Problem

Objective Value: \$30

License	A	B	C				
Minimize	$1s_A$	$1s_B$	$1s_C$				
Bid 1	s_A	s_B	0			=	\$22
Bid 2	0	s_B	s_C	δ_2		↓	\$24
Bid 3	s_A	0	s_C	δ_3		↓	\$20
Bid 4	s_A	s_B	0	δ_4		↓	\$16
Bid 5	0	0	s_C			=	\$8

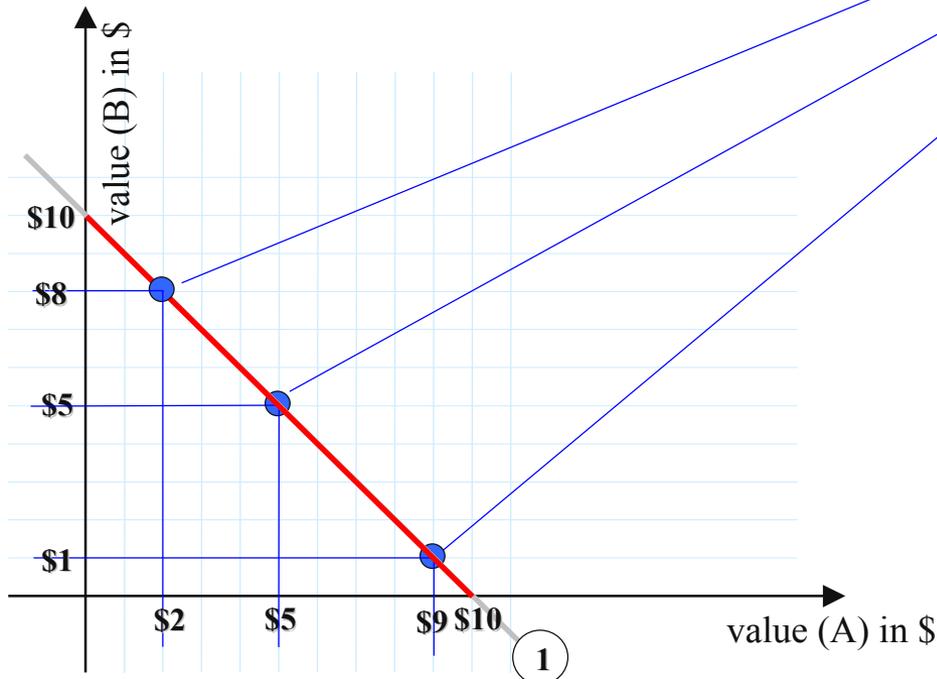
Variables	9	, 13	, 8	↓	0
Variables	3	, 3	, 0	↓	0



Alternative Shadow Prices

Bid	1
Bid amt.	\$10
Package	AB
Solution	1

License	Shadow Price
A	2
B	3

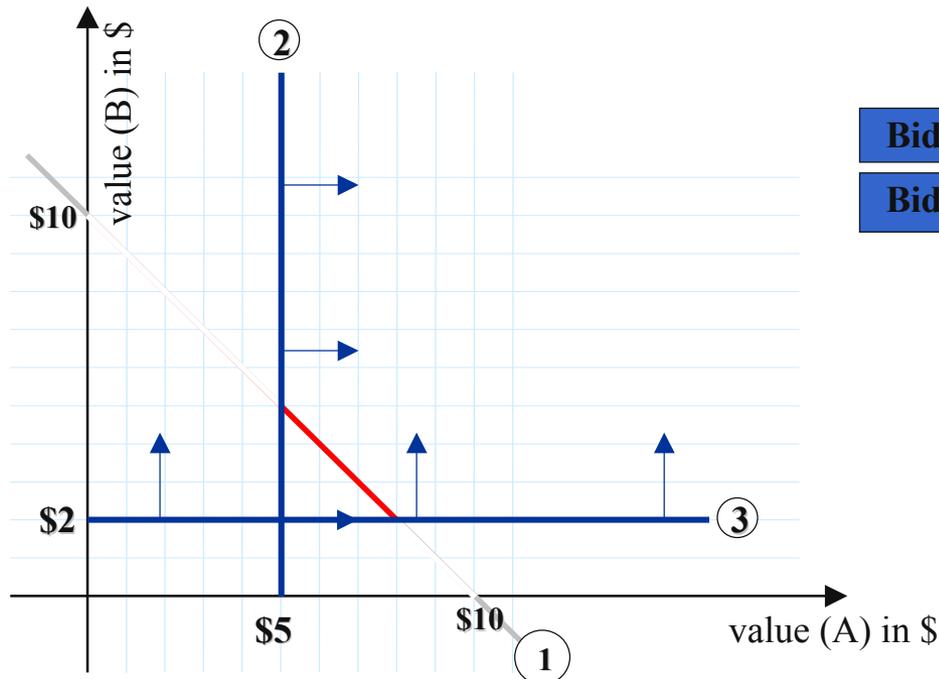




Alternative Shadow Prices

Bid	1	2	3
Bid amt.	\$10	\$5	\$2
Package	AB	A	B
Solution	1	0	0

License	Shadow Price
A	
B	



Dual Constraints

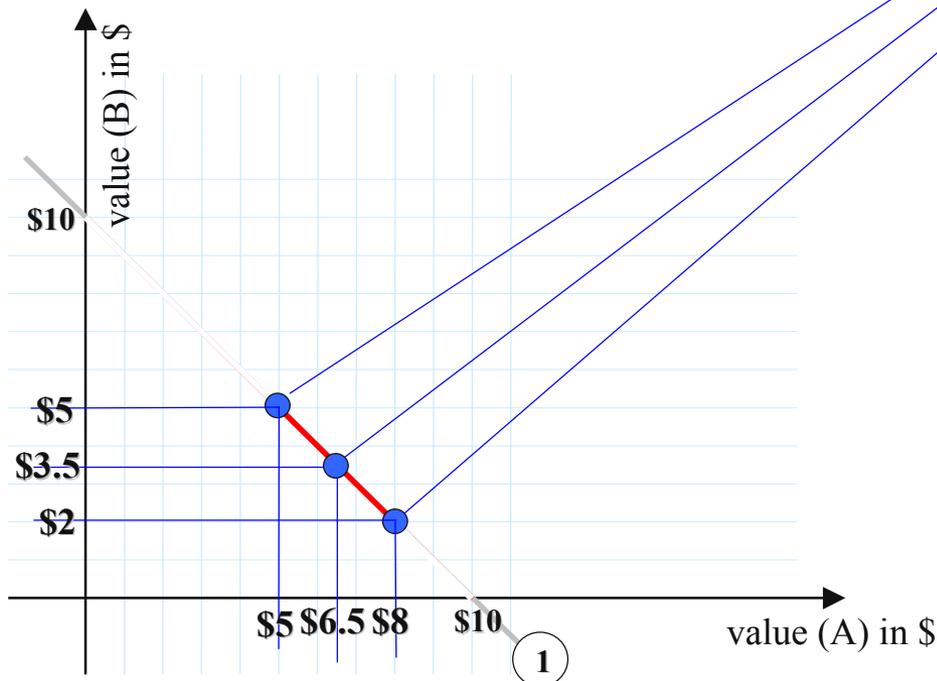
	A		B		
Bid 2	<input type="text"/>	+	<input type="text"/>	\geq	\$5
Bid 3	<input type="text"/>	+	<input type="text"/>	\geq	\$2



Alternative Shadow Prices

Bid	1	2	3
Bid amt.	\$10	\$5	\$2
Package	AB	A	B
Solution	1	0	0

License	Shadow Price
A	6.5
B	3.5



Solution: Uses a centering algorithm to choose among alternative optima



Alternatives

- ▲ DeMartini, Kwasnica, Ledyard and Porter (1999) recommend minimizing the maximum difference that a shadow price might be off from the largest bid price on that bid and performing these calculations sequentially
- ▲ Rassenti, Smith and Bulfin (1982) suggest solving series of optimization problems to determine those bids that will always be in the solution, those that will never be in the solution and those that have the potential to be in some optimal solutions
- ▲ Our testing has suggested an alternative calculation: Determine the max and min possible value for each shadow price and then choose a solution at the center of this hyper-rectangle



Shadow Prices Induce Ties

- ▲ Identical shadow prices for all bidders will lead to many ties, which cause computational problems when mutual exclusive bid rules exist
 - Ties make it harder to solve for the provisional winners
 - Tie-breaking procedure is computationally challenging when many ties exist
- ▲ Possible Solutions
 - Computational stages where ties are not calculated early in auction (when the number of ties will likely be in the millions)
 - Bidder-specific MABs that come from a combination of generic shadow prices and a value determined by the “quality of a bidder’s previous bids” (Milgrom; Plott; Sandholm, Suri, Gilpen, and Levine)

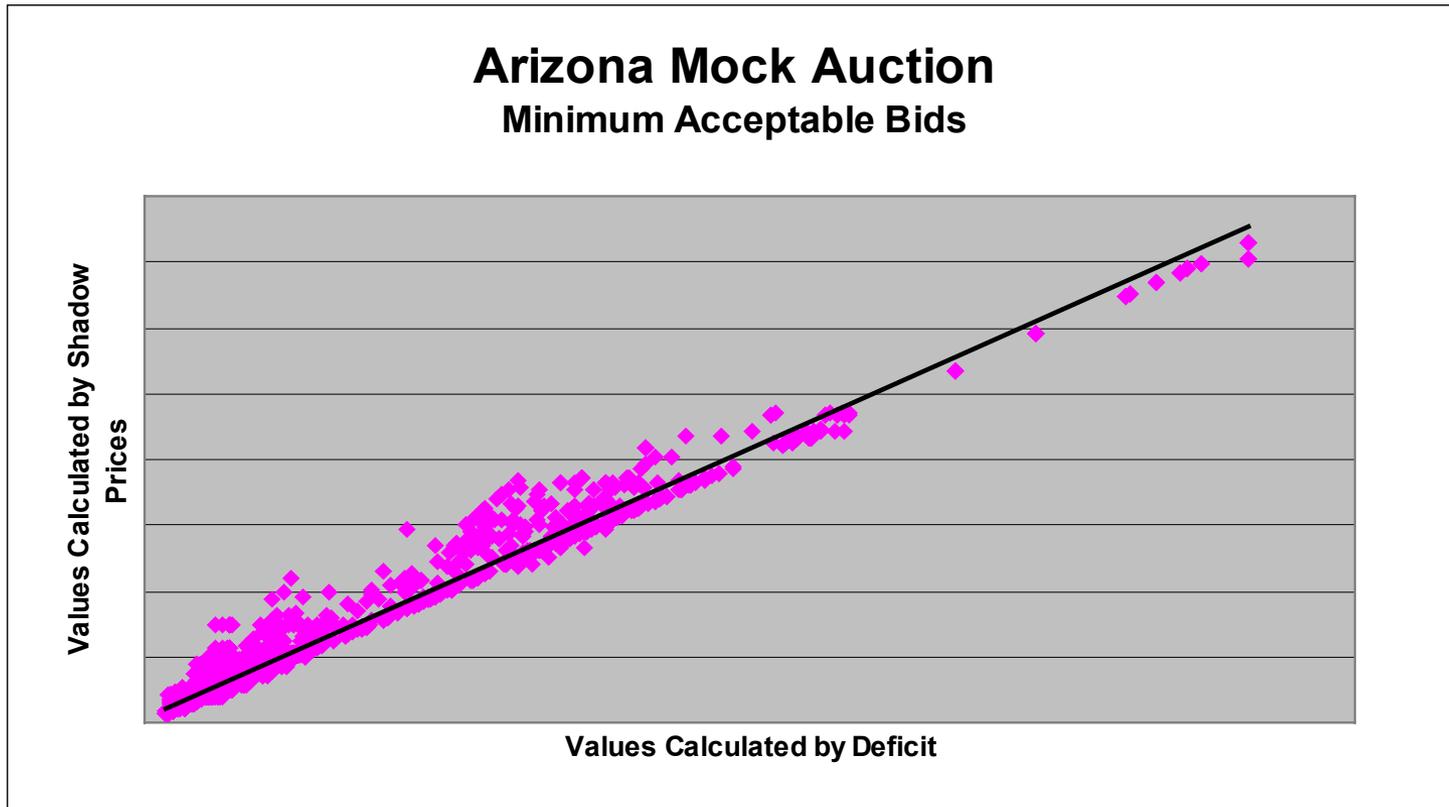


A Merging of Two Methods: Deficit and Shadow Prices

- ▲ Deficit calculations are careful, conservative estimates of what it would take for a bid to be in a provisionally winning bid set
 - However, they will take too long when there are many eligible bidders placing many new bids
- ▲ Computational stages for MAB calculations
 - In early rounds, use approximation -- shadow prices
 - In later rounds, use deficit calculations
 - ▲ May still not be able to do a deficit calculation for every license and package of every eligible bidder
 - ▲ Possible alternative: Perform deficit calculations on recent bids for each bidder



Deficit versus Shadow Prices



MABs calculated from Shadow Prices
resemble deficit MABs



Outline

- ▲ Review of Auction #31 rules impacting scalability and possible alternatives
 - Determining Max Revenue each round
 - Choosing among ties
 - Minimum Acceptable Bid Price
- ▲ Mechanism for testing scalability -- BidBots
- ▲ Test design
- ▲ Ideas on where to go from here



Testing Mechanism

- ▲ BidBots - Automated Auction Simulation Program
 - Rule-based intelligent agents to model bidders placing bids each round
 - Impose auction rules
 - Vary size of auction
 - ▲ Licenses
 - ▲ Bidders
 - ▲ Packages per Bidder
- ▲ Sun Solaris machine with 4 UltraSparc-II 450MHz CPUs and 2 GB of memory



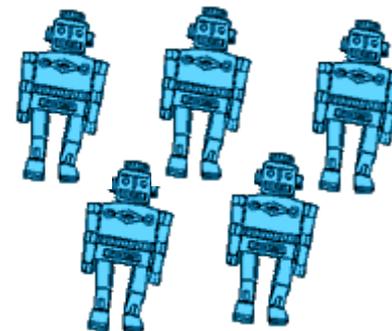
BidBots Software

- ▲ Fundamental purposes include
 - Primary: Provide large data sets to test a large-scale solver
 - Secondary: Provide a means to analyze effects of auction rules on solvability
- ▲ Realized advantages
 - Relieve burden of human participation
 - ▲ Time requirements
 - ▲ No user interface requirements
 - ▲ Bidders do not complain about not winning, do not need lunch breaks, and are willing to work long hours
 - Perform “As Fast As It Can Run” testing
 - ▲ Bidding takes less than 1 second per active bidder
 - ▲ Multiple auctions can be run in a day



BidBot Characteristics

- ▲ Provides basic bidder functionality
 - Place new bids (with increments)
 - Renew previous bids
 - Dynamically create new packages
 - Utilize activity waivers
 - Reduce eligibility
- ▲ Bidders are given budget constraints
- ▲ Eligibility based on historic data
 - Many small bidders with some large bidders
- ▲ Adjacency matrix for geographic synergies that determine the value of packages
- ▲ Value and straightforward bidders with random increment bidding based on historical data





Outline

- ▲ Review of Auction #31 rules impacting scalability and possible alternatives
 - Determining Max Revenue each round
 - Choosing among ties
 - Minimum Acceptable Bid Price
- ▲ Mechanism for testing scalability -- BidBots
- ▲ Test design
- ▲ Ideas on where to go from here



Test Auction Design

For each of these alternatives, answer the question:

How large an auction can be run?

1. Exact optimization calculations for all aspects of the auction

Staging:

2. Early in the auction, use shadow prices instead of deficit calculations to determine minimum acceptable bid
3. Early in the auction, use shadow prices and do not choose among ties, i.e. take whatever solution obtained in Maximum Revenue calculation
4. Early in the auction, use shadow prices, do not break ties, and stop optimization after x minutes of calculation or y percent of optimality
5. Other changes to rules

At later stage of auction -- Always do everything exactly



Testing Characteristics

- ▲ Currently using all default parameters of CPLEX 7.1
- ▲ Testing will include fine tuning the settings of the solver
- ▲ Could use more than one machine and more than one solving strategy, stopping when first machine finds optimal solution
- ▲ Testing will likely include special structure implementation



Where do we go next?

- ▲ More testing using both BidBots and mock auctions
- ▲ Shadow price testing:
 - What is the best way to adjust shadow prices?
 - What happens when licenses have few or no bidders?
 - How can we obtain a bidder-specific MAB that uses shadow prices?
- ▲ Click-box bidding versus greater flexibility in submission of bids:
 - Does providing more digits of accuracy help to eliminate ties?
 - Issues of transparency, speed of auction, collusion
- ▲ When should computational staging occur?

Others Issues?