

Generating Trading Agent Strategies:

Analytic and Empirical Methods for Infinite and Large Games

Daniel Reeves

2006 Feb 23

Coauthors:

Michael Wellman, Jeffrey MacKie-Mason,

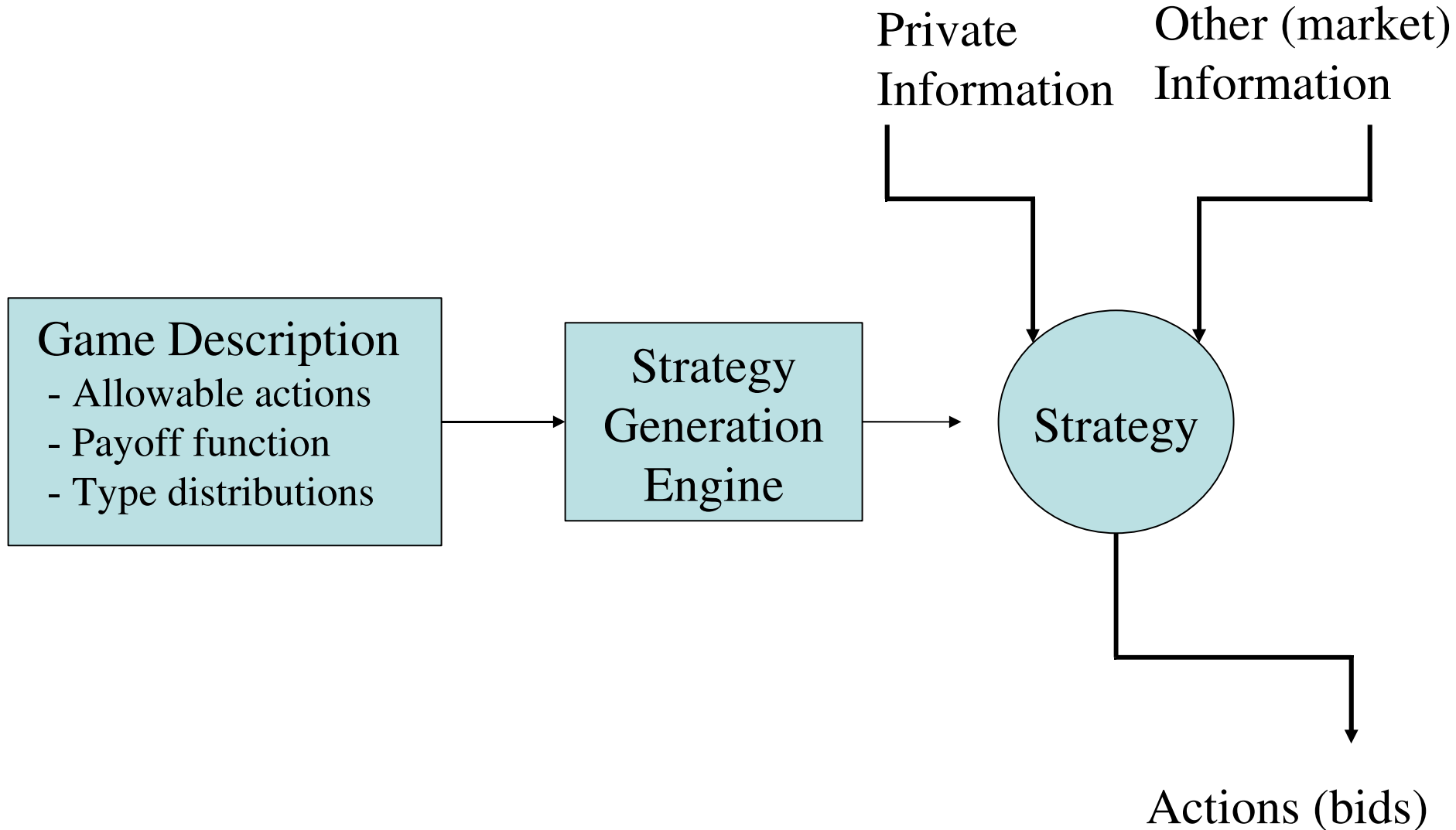
Anna Osepayshvili, Kevin Lochner,

Shih-Fen Cheng, Rahul Suri,

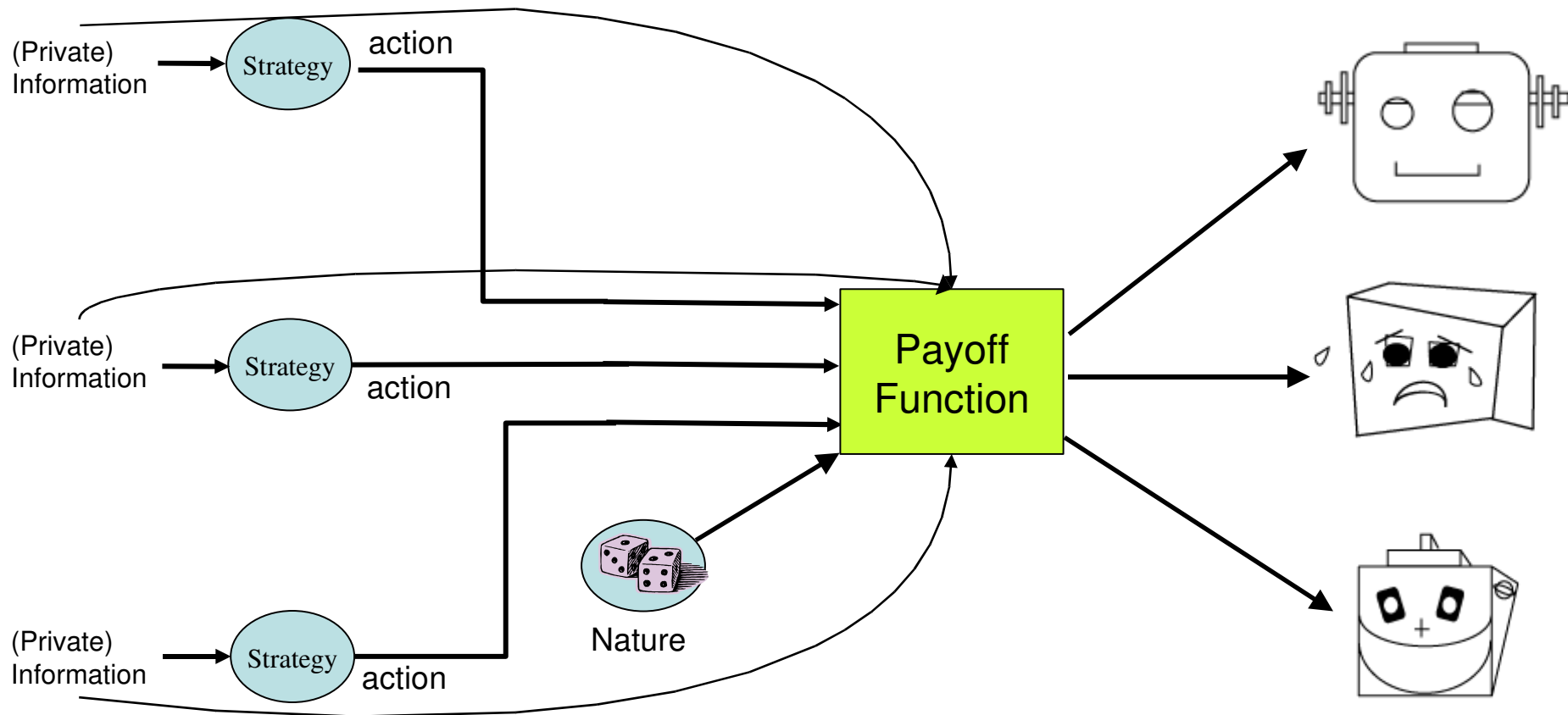
Yevgeniy Vorobeychik, and Maxim Rytin



Motivation



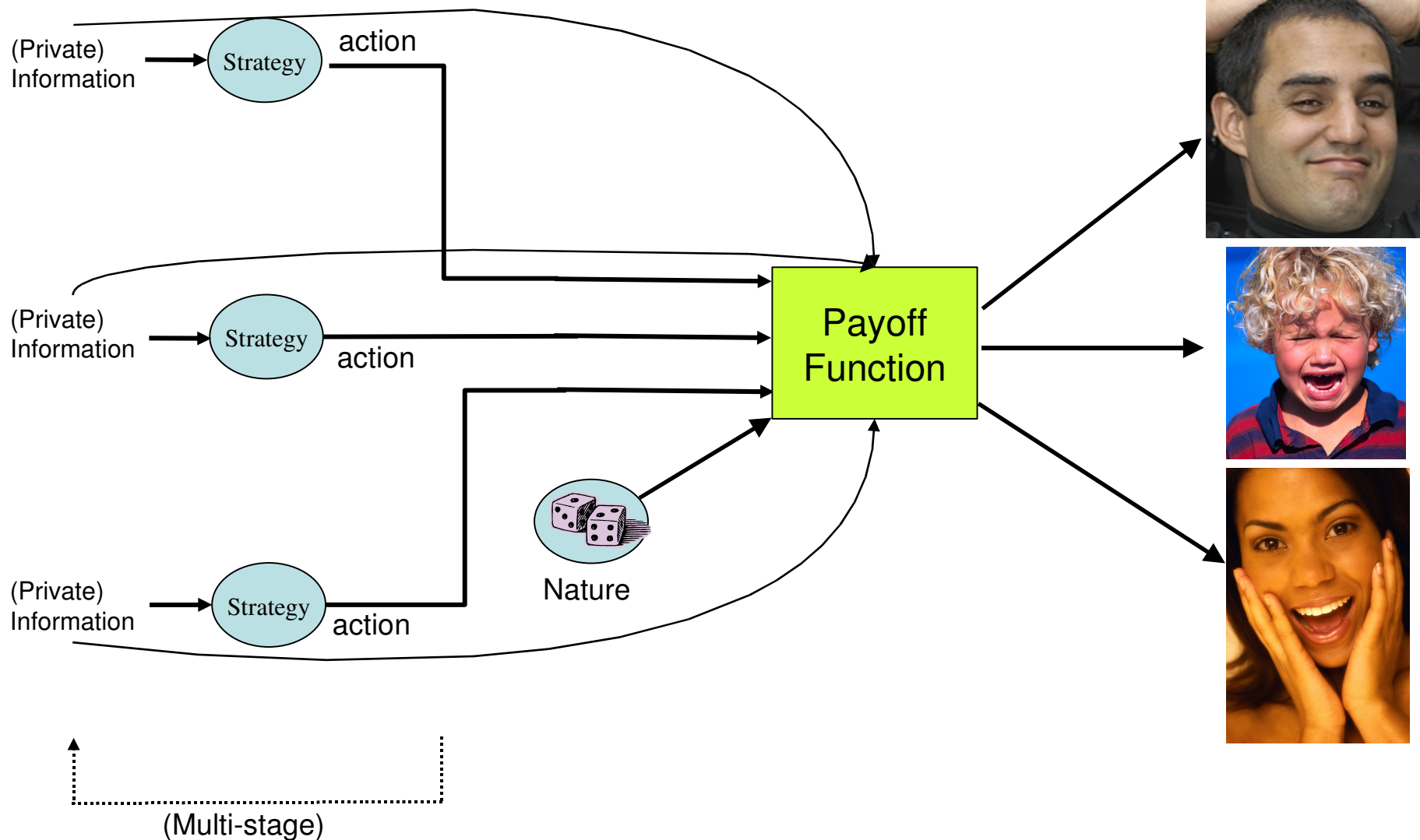
Game Theory Primer



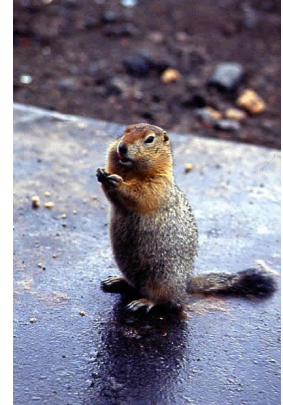
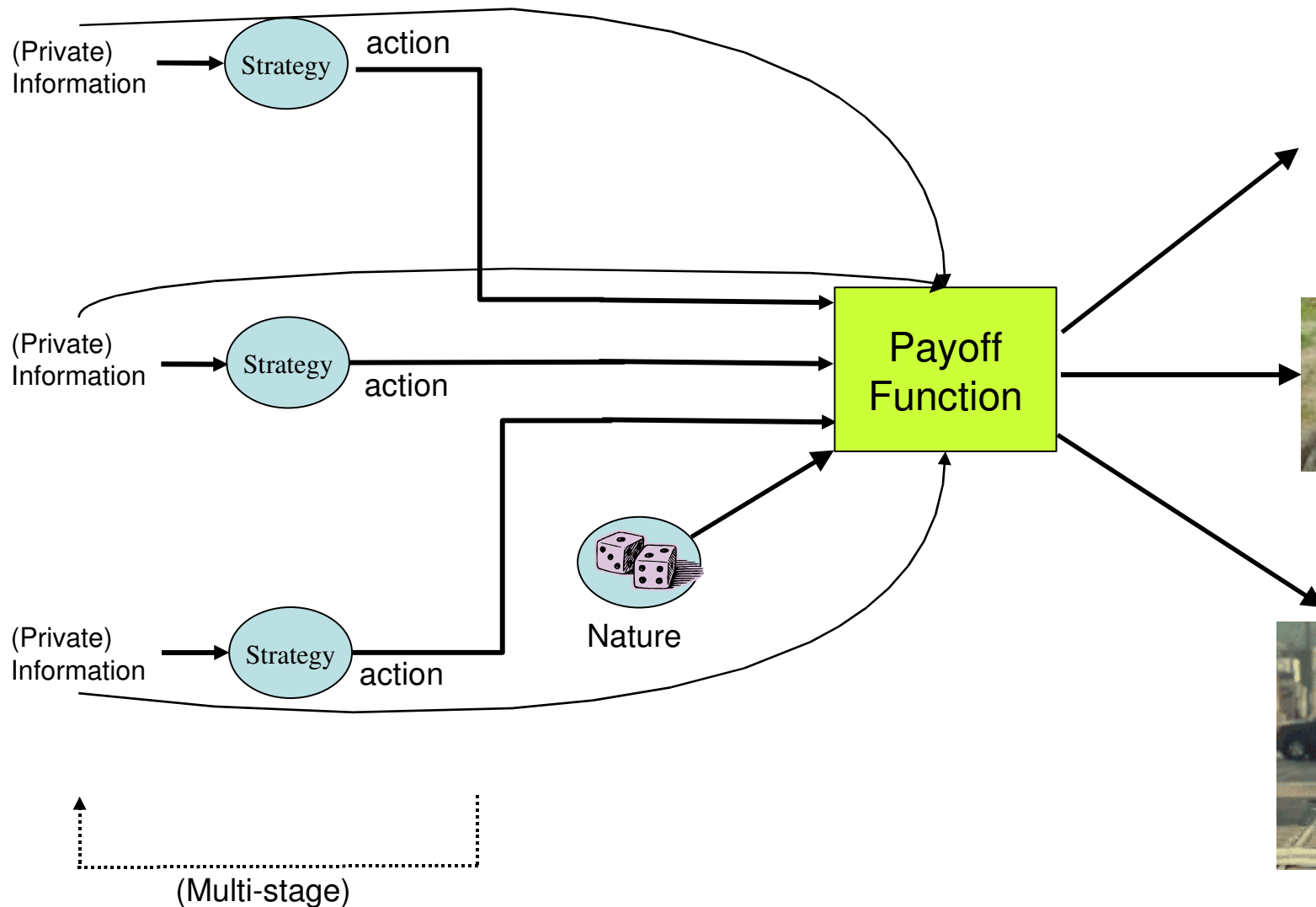
↑

(Multi-stage)

Game Theory Primer



Game Theory Primer



Game Theory Primer 2

- *Best-Response Strategy* = optimal strategy given known strategies of the other players
- *Nash Equilibrium* = profile of strategies such that each strategy is a best response to the others
- *Bayes-Nash Equilibrium* = generalization of NE to the case of incomplete information, for expected-utility maximizing players
- *Normal-form Game* = defined in terms of strategies
- *Symmetric Game* = no distinct player roles (except Nature)
- *Epsilon of a Profile* = best gain from deviating (0 iff Nash)

Outline

- Best-response strategies in one-shot, 2-player, infinite games of incomplete information
- Empirical game methodology for multi-stage, multi-player games
("Taming Monster Games")
- Taming 2 particular monster games:
 - Simultaneous Ascending Auctions (SAA)
 - Trading Agent Competition (TAC Travel)

Our Class of Infinite Games

- 2-player, one-shot, infinite games of incomplete information
- Piecewise uniform type distributions
- Payoff functions of the form:

$$u(t, a, t', a') = \begin{cases} \theta_1 t + \rho_1 a + \theta'_1 t' + \rho'_1 a' + \phi_1 & \text{if } -\infty < a + \alpha a' < \beta_2 \\ \theta_2 t + \rho_2 a + \theta'_2 t' + \rho'_2 a' + \phi_2 & \text{if } \beta_2 \leq a + \alpha a' \leq \beta_3 \\ \dots & \\ \theta_I t + \rho_I a + \theta'_I t' + \rho'_I a' + \phi_I & \text{if } \beta_I \leq a + \alpha a' \leq +\infty \end{cases}$$

Games in our Class

Game	θ	ρ	θ'	ρ'	ϕ	β	α
FPSB	$0, 1/2, 1$	$0, -1/2, -1$	$0, 0, 0$	$0, 0, 0$	0	$0, 0$	-1
Vickrey Auction	$0, 1/2, 1$	$0, 0, 0$	$0, 0, 0$	$0, -1/2, -1$	0	$0, 0$	-1
Vicious Vickrey Auction	$0, \frac{1-k}{2}, 1-k$	$k, k/2, 0$	$-k, -k/2, 0$	$0, \frac{k-1}{2}, k-1$	0	$0, 0$	-1
Supply Chain Game	$-1, -1, 0$	$1, 1, 0$	$0, 0, 0$	$0, 0, 0$	0	v, v	1
Bargaining Game (seller)	$-1, -1, 0$	$1-k, 1-k, 0$	$0, 0, 0$	$k, k, 0$	0	$0, 0$	-1
(buyer)	$0, 1, 1$	$0, -k, -k$	$0, 0, 0$	$0, 1-k, 1-k$	0	$0, 0$	-1
All-Pay Auction	$0, 1/2, 1$	$-1, -1, -1$	$0, 0, 0$	$0, 0, 0$	0	$0, 0$	-1
War of Attrition	$0, 1/2, 1$	$-1, -1/2, 0$	$0, 0, 0$	$0, -1/2, -1$	$0, 0, 0$	$0, 0$	-1
* Shared-Good Auction	$0, 1/2, 1$	$0, -1/4, -1/2$	$0, 0, 0$	$1/2, 1/4, 0$	0	$0, 0$	-1
* Joint Purchase Auction	$0, 1$	$0, -1/2$	$0, 0$	$0, 1/2$	$0, -C/2$	C	1
Subscription Game	$0, 1$	$0, -1$	$0, 0$	$0, 0$	$0, 0$	C	1
Contribution Game	$0, 1$	$-1, -1$	$0, 0$	$0, 0$	$0, 0$	C	1

$$u(t, a, t', a') =$$

$$\left\{ \begin{array}{ll} \theta_1 t + \rho_1 a + \theta'_1 t' + \rho'_1 a' + \phi_1 & \text{if } -\infty < a + \alpha a' < \beta_2 \\ \theta_2 t + \rho_2 a + \theta'_2 t' + \rho'_2 a' + \phi_2 & \text{if } \beta_2 \leq a + \alpha a' \leq \beta_3 \\ \dots & \\ \theta_I t + \rho_I a + \theta'_I t' + \rho'_I a' + \phi_I & \text{if } \beta_I \leq a + \alpha a' \leq +\infty \end{array} \right.$$

Piecewise Linear Strategies

$$s(t) = \begin{cases} m_1 t + b_1 & \text{if } -\infty < t \leq c_2 \\ m_2 t + b_2 & \text{if } c_2 < t \leq c_3 \\ \dots & \\ m_{K-1} t + b_{K-1} & \text{if } c_{K-1} < t \leq c_K \\ m_K t + b_K & \text{if } c_K < t \leq +\infty, \end{cases}$$

- Specified by the vectors **c**, **m**, **b**

Existence and Computation of Piecewise Linear Best Responses

- Theorem 1: Given a payoff function with I regions, an opponent type distribution with cdf F that is piecewise uniform with J pieces, and a piecewise linear strategy function with K pieces, the best response is itself a piecewise linear function with no more than $2(I-1)(J+K-2)$ piece boundaries.

Proof Sketch

- For arbitrary own type t , and opponent type a random variable T , find own action a maximizing $E_T[u(t, a, T, s(T))]$
- (Numerical maximization not applicable due to parameter t)
- Above works out to be a piecewise polynomial in a (parameterized by t)
- For given t , finding optimal a is straightforward
- Remains to find partitioning of type space such that within each type range, optimal action is a linear function of t
- This can be done in polynomial time

Example: First-Price Sealed Bid Auction (FPSB)

- Types (valuations) drawn from $U[0,1]$
- Payoff function:

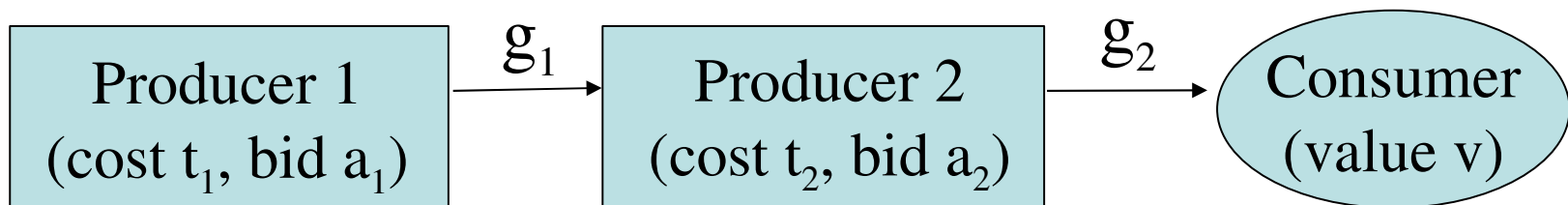
$$u(t, a, a') = \begin{cases} t - a & \text{if } a > a' \\ (t - a)/2 & \text{if } a = a' \\ 0 & \text{otherwise.} \end{cases}$$

- Known Bayes-Nash equilibrium:
 $a(t)=t/2$ (Vickrey, 1961)
- Found in as few as one iteration from a variety of seed strategies

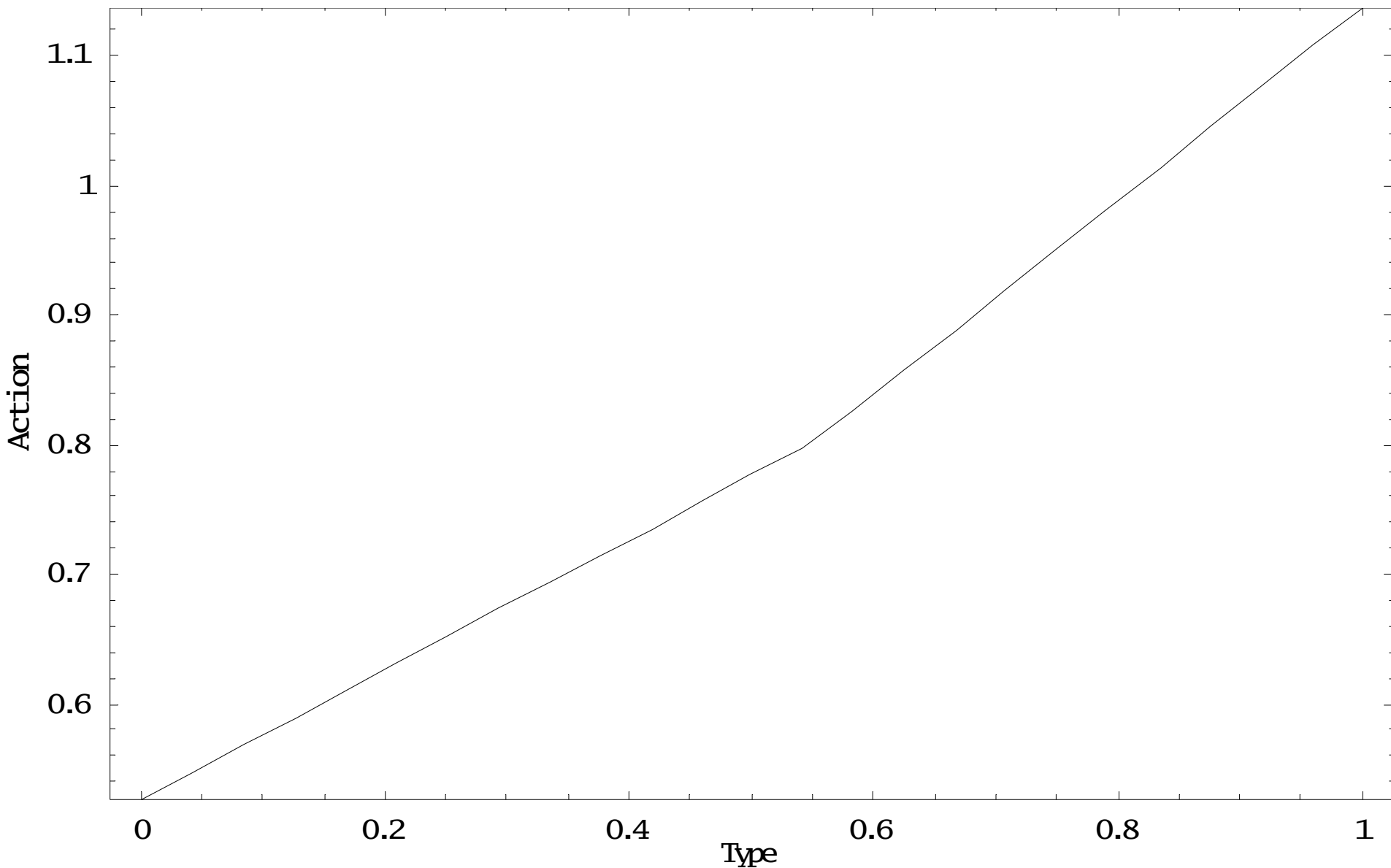
Example: Supply-chain Game

- Producers' Costs $U[0,1]$
- Consumer's Valuation v (known)
- Payoff function:

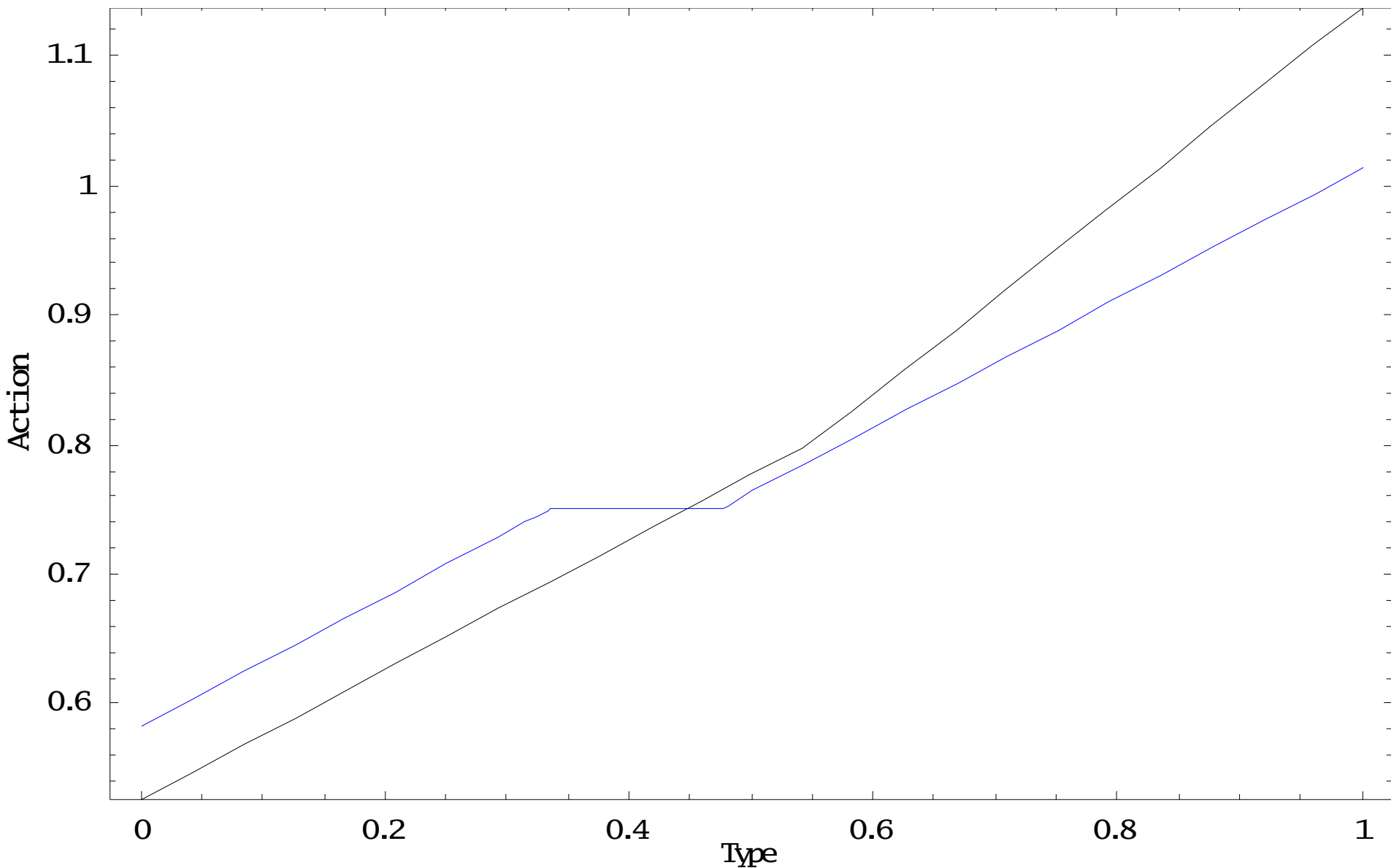
$$u(t_1, a_1, a_2) = \begin{cases} a_1 - t_1 & \text{if } a_1 + a_2 \leq v \\ 0 & \text{otherwise} \end{cases}$$



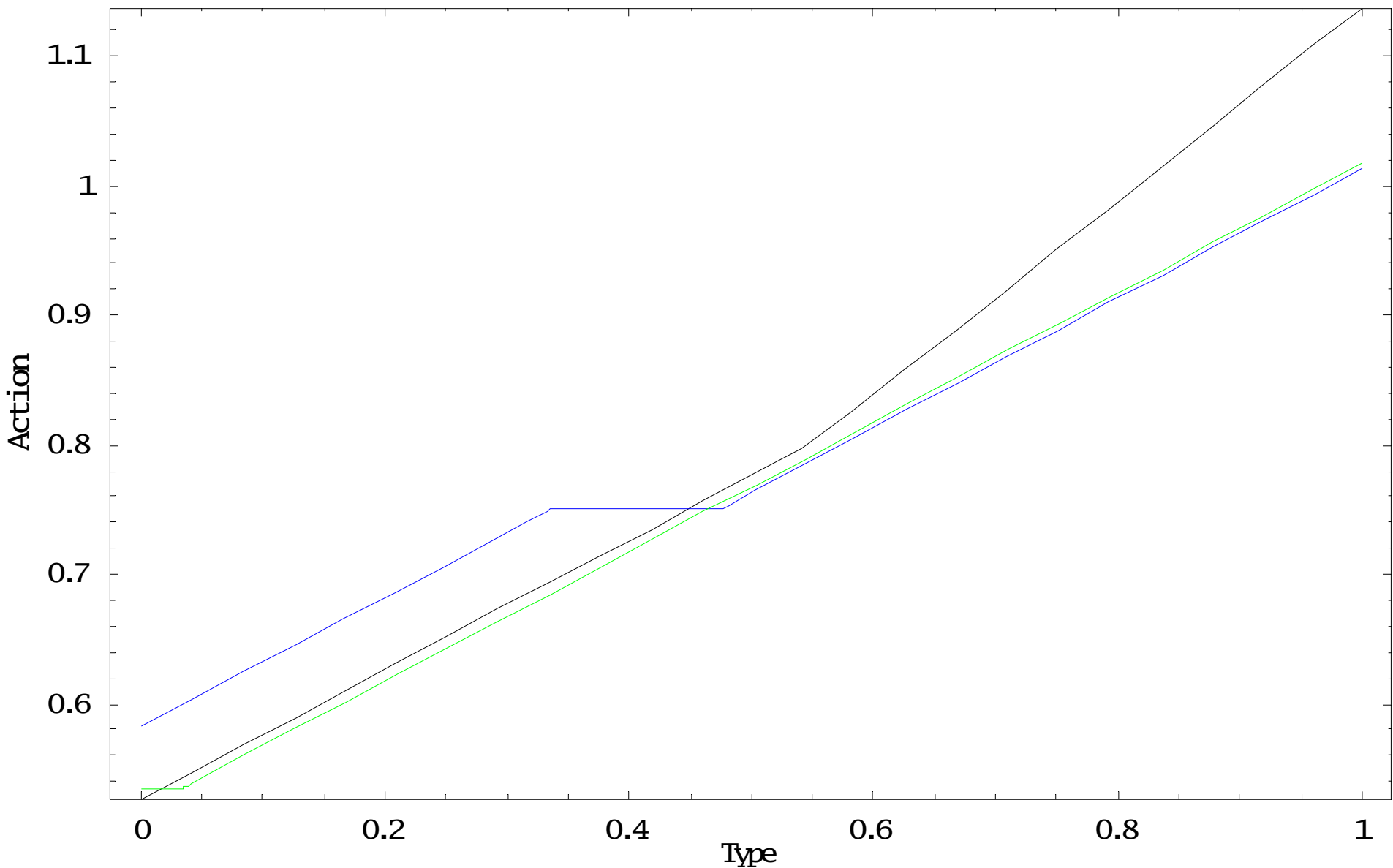
Finding Best Responses



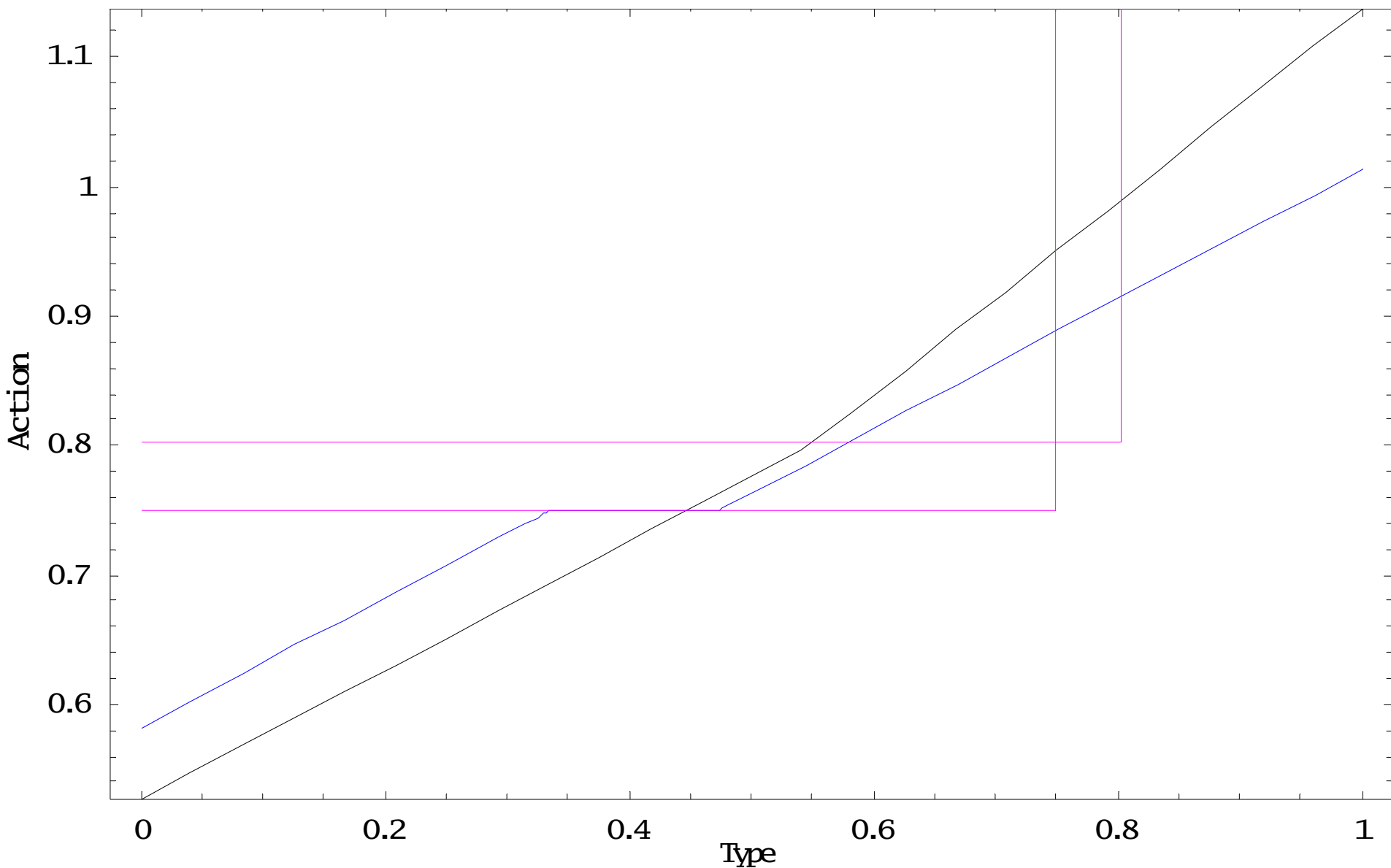
Finding Best Responses



Finding Best Responses



Finding Best Responses



Example: Bargaining Game

- (aka, sealed-bid k-double auction)
- Buyer and seller place bids, transaction happens iff they overlap
- Transaction price is some linear combination of the bids
- Known equilibrium (Chatterjee & Samuelson, 1983) with $k=1/2$ for seller (1) and buyer (2):

$$a_1(t_1) = 2/3t_1 + 1/4$$

$$a_2(t_2) = 2/3t_2 + 1/12$$

- Found in several iterations from truthful bidding

Example: Joint Purchase Auction

- Variants: contribution/subscription games
- 2 agents want to jointly acquire a good costing C
- Mechanism: simultaneously offer contributions; buy iff $\text{sum} > C$ and split the excess ($\text{sum} - C$) evenly
- Nash: $\frac{2}{3}t + \frac{C}{4} - \frac{1}{6}$

Example: Shared-Good Auction

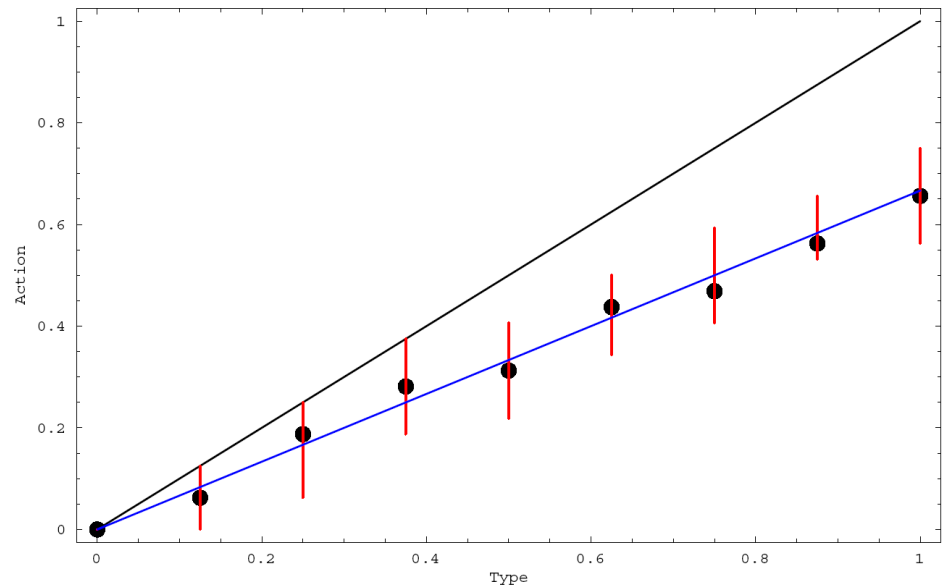
- New mechanism, similar to the divorce-settlement game; undoes joint-purchase
- Agents place bids for a good they currently share, valuations $\sim U[A, B]$
- High bidder gets the good and pays half its bid to the low bidder in compensation

$$u(t, a, a') = \begin{cases} t - a/2 & \text{if } a > a' \\ a'/2 & \text{otherwise} \end{cases}$$

Equilibrium in Shared-Good Auction

$$a(t) = \frac{2t + A}{3}$$

- Found in one iteration from truthful bidding (for any specific $[A,B]$)



Example: Vicious Vickrey Auction

- Generalization of a Vickrey Auction (Brandt & Weiss, 2001) to allow for disutility from opponent's utility (eg, business competitors)

$$u(t, a, t', a') = \begin{cases} (1 - k)(t - a') & \text{if } a > a' \\ ((1 - k)(t - a') - k(t' - a))/2 & \text{if } a = a' \\ -k(t' - a) & \text{otherwise} \end{cases}$$

- Brandt & Weiss consider only the complete information version

Equilibrium in Vicious Vickrey

- $a(t) = (k+t)/(k+1)$
- Reduces to truthful bidding for the standard Vickrey Auction ($k=0$)
- Iterated best-response solver finds this equilibrium (for specific values of k) within several iterations from a variety of seed strategies

Conclusions:

Best-Response Solver

- First algorithm for finding best-response strategies in a broad class of infinite games of incomplete information
- Confirms known equilibria (eg, FPSB), confirms equilibria we derive here (Supply-Chain game), discovers equilibria in new games (eg, Joint Purchase and Shared-good auction)
- Goal: characterize the class of games for which iterated best-response converges

Taming Monster Games: Overview

- Determining candidate strategies
- Game simulators and brute-force estimation
- Variance reduction for Monte Carlo Sampling
 - Control Variates
 - (Quasi-Random Sampling, Importance Sampling)
- **Player Reduction**
- Analyzing Empirical Games
 - Gambit, Amoeba, Replicator Dynamics
- Sensitivity Analysis
 - PM distributions
 - Confidence bounds on equilibria
- Killer App: Trading Agent Competition (TAC)

Reprise: First-Price Sealed-Bid Auction (FPSB)

- Types (valuations) drawn from $U[0,1]$
- Payoff function (2-player case):

$$u(t, a, a') = \begin{cases} t - a & \text{if } a > a' \\ (t - a)/2 & \text{if } a = a' \\ 0 & \text{otherwise.} \end{cases}$$

- Strategy space is set of functions from type to action
- Known Bayes-Nash equilibrium: $a(t) = (n-1)/n t$

FPSB_n

- Start with a baseline strategy
 - Truthful bidding
- Generalize via parameters
 - Shade factor
 - (Translational parameter, $kt+b$, etc)
- Restricted game:
 - For all agents i in $\{1, \dots, n\}$, bid $k_i t$ for k_i in $[0, 1]$
- Further restrict the game by discretizing k

Theoretical Results for FPSBn

- Expected Payoff for playing k_i against everyone else playing k :

$$u_i(k_i, k) = \begin{cases} \frac{1}{2n} & \text{if } k_i = k = 0 \\ \frac{1-k_i}{n+1} \left(\frac{k_i}{k}\right)^{n-1} & \text{if } k_i \leq k \\ \frac{(1-k_i)((n-1)k_i^2 - (n-1)k^2)}{2(n+1)k_i^2} & \text{otherwise.} \end{cases}$$

- Best response to everyone else playing k :

$$BR(k) \equiv \arg \max_{k_i} u_i(k_i, k) = \begin{cases} \text{undefined} & \text{if } k = 0 \\ \xi & \text{if } k < \frac{n-1}{n} \\ \frac{n-1}{n} & \text{if } k \geq \frac{n-1}{n} \end{cases}$$

Theoretical Results for FPSBn

- Expected Payoff for playing k_i against everyone else playing k :

$$u_i(k_i, k) = \begin{cases} \frac{1}{2n} & \text{if } k_i = k = 0 \\ \frac{1-k_i}{n+1} \left(\frac{k_i}{k}\right)^{n-1} & \text{if } k_i \leq k \\ \frac{(1-k_i)((n-1)k_i^2 - (n-1)k^2)}{2(n+1)k_i^2} & \text{otherwise.} \end{cases}$$

- Best response to everyone else playing k :

$$\frac{\sqrt[3]{3} \left(k^2 (n^2 - 1) \left(9n + \sqrt{3(n+1)((n-1)k^2 + 27(n+1))} + 9 \right) \right)^{2/3} - 3^{2/3} k^2 (n^2 - 1)}{3(n+1) \sqrt[3]{k^2 (n-1) \left(9n^2 + 18n + (n+1)^{3/2} \sqrt{3(n-1)k^2 + 81(n+1)} + 9 \right)}}$$

Theoretical Results for FPSBn

- Expected Payoff for playing k_i against everyone else playing k :

$$u_i(k_i, k) = \begin{cases} \frac{1}{2n} & \text{if } k_i = k = 0 \\ \frac{1-k_i}{n+1} \left(\frac{k_i}{k}\right)^{n-1} & \text{if } k_i \leq k \\ \frac{(1-k_i)((n-1)k_i^2 - (n-1)k^2)}{2(n+1)k_i^2} & \text{otherwise.} \end{cases}$$

- Best response to everyone else playing k :

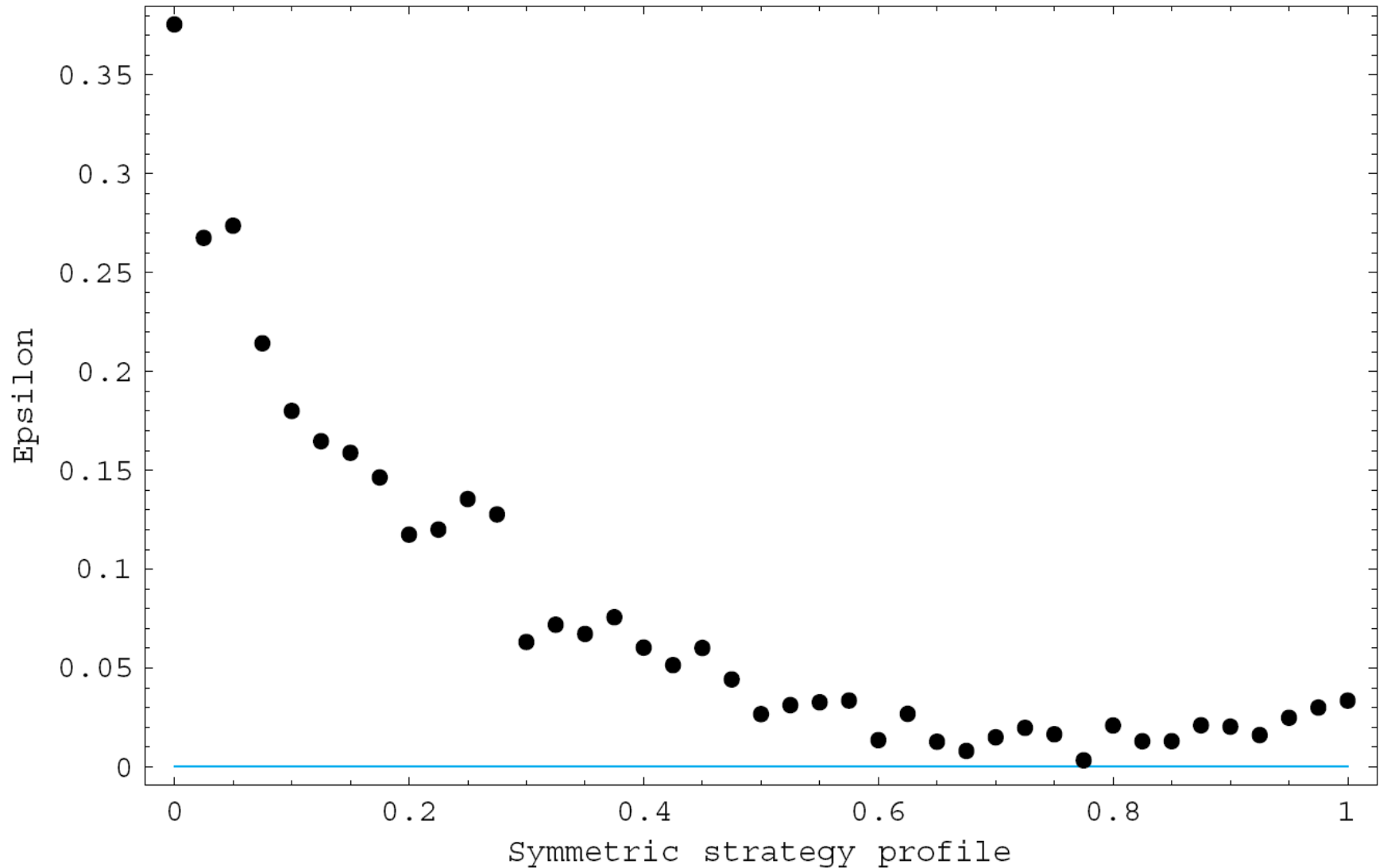
$$BR(k) \equiv \arg \max_{k_i} u_i(k_i, k) = \begin{cases} \text{undefined} & \text{if } k = 0 \\ \xi & \text{if } k < \frac{n-1}{n} \\ \frac{n-1}{n} & \text{if } k \geq \frac{n-1}{n} \end{cases}$$

Epsilon Metric for FPSBn

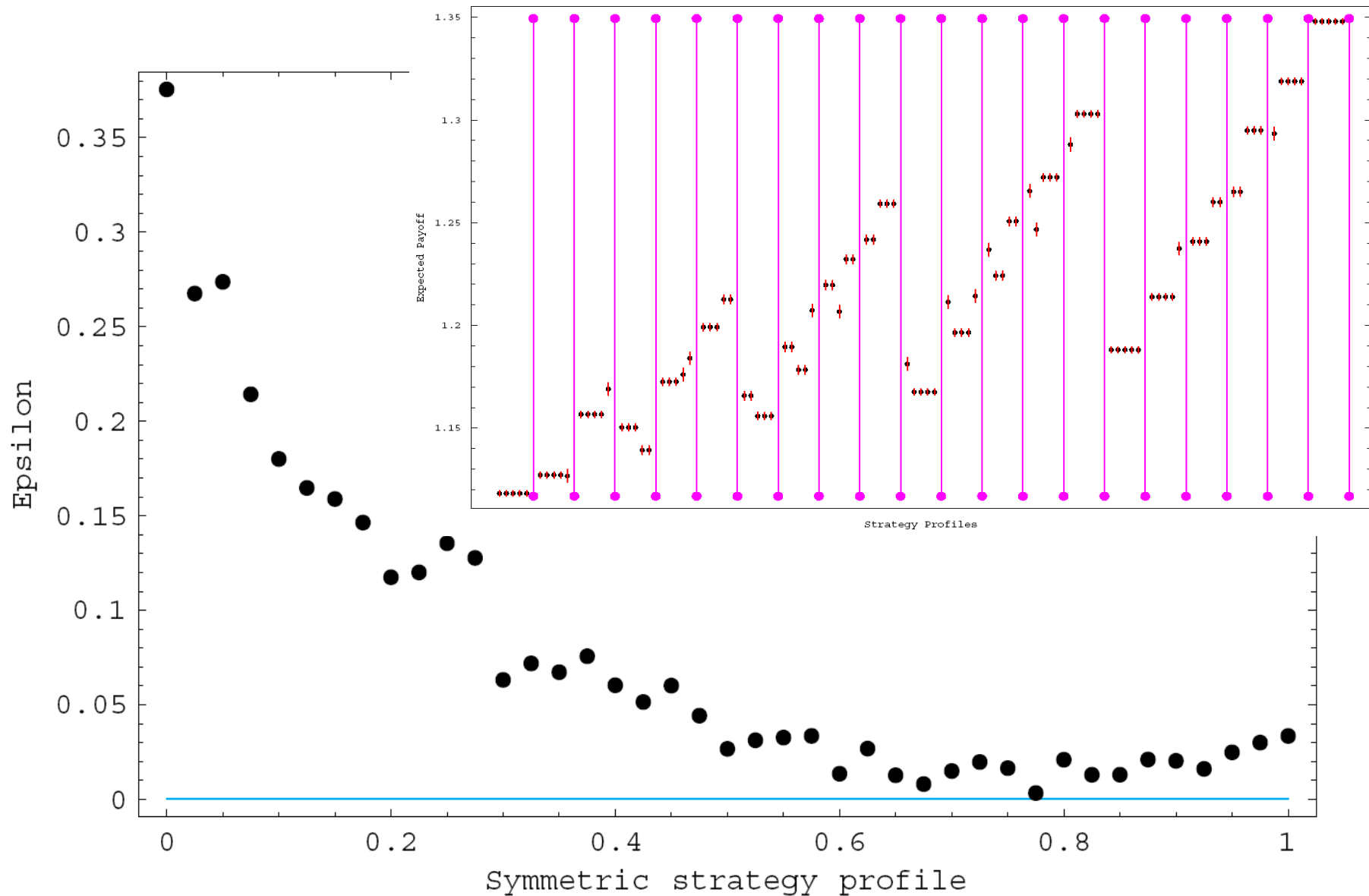
- The epsilon for a profile s is greatest possible gain from deviating (payoff of best unilateral deviation from s minus payoff in s)
- For a symmetric profile (k) in FPSBn:

$$\varepsilon_{FPSBn}(k) = \max_x (u_i(x, k)) - u_i(k, k) = \begin{cases} \frac{1}{2} - \frac{1}{2n} & \text{if } k = 0 \\ \frac{(k - \xi)(\xi^2 - k\xi + \xi + k + (\xi - 1)(\xi + k)n)}{2\xi^2(n + 1)} & \text{if } k < \frac{n - 1}{n} \\ \frac{1 - n + k \left(\left(\frac{n - 1}{kn} \right)^n + n - 1 \right)}{n^2 - 1} & \text{otherwise.} \end{cases}$$

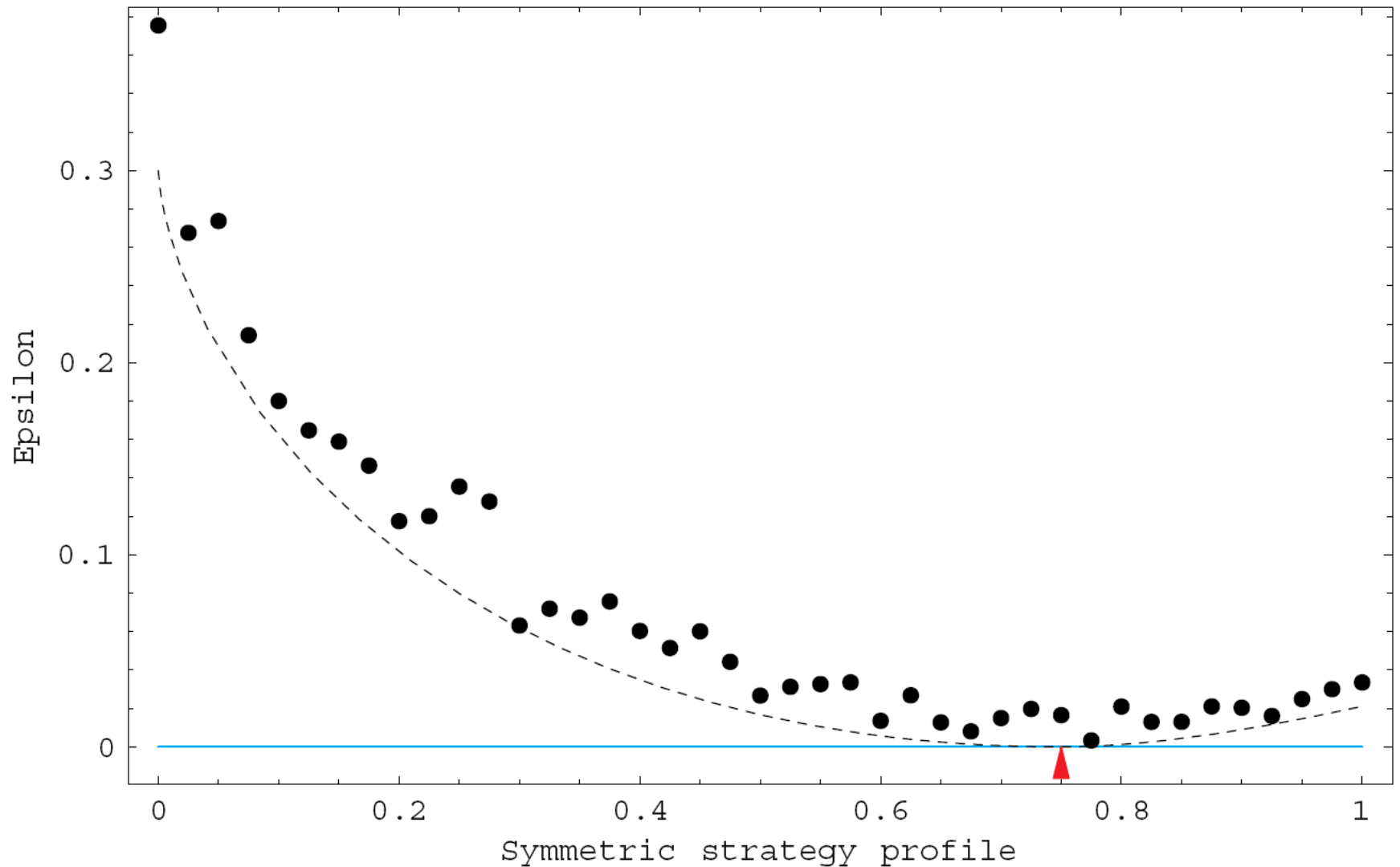
Game Simulation: FPSB4



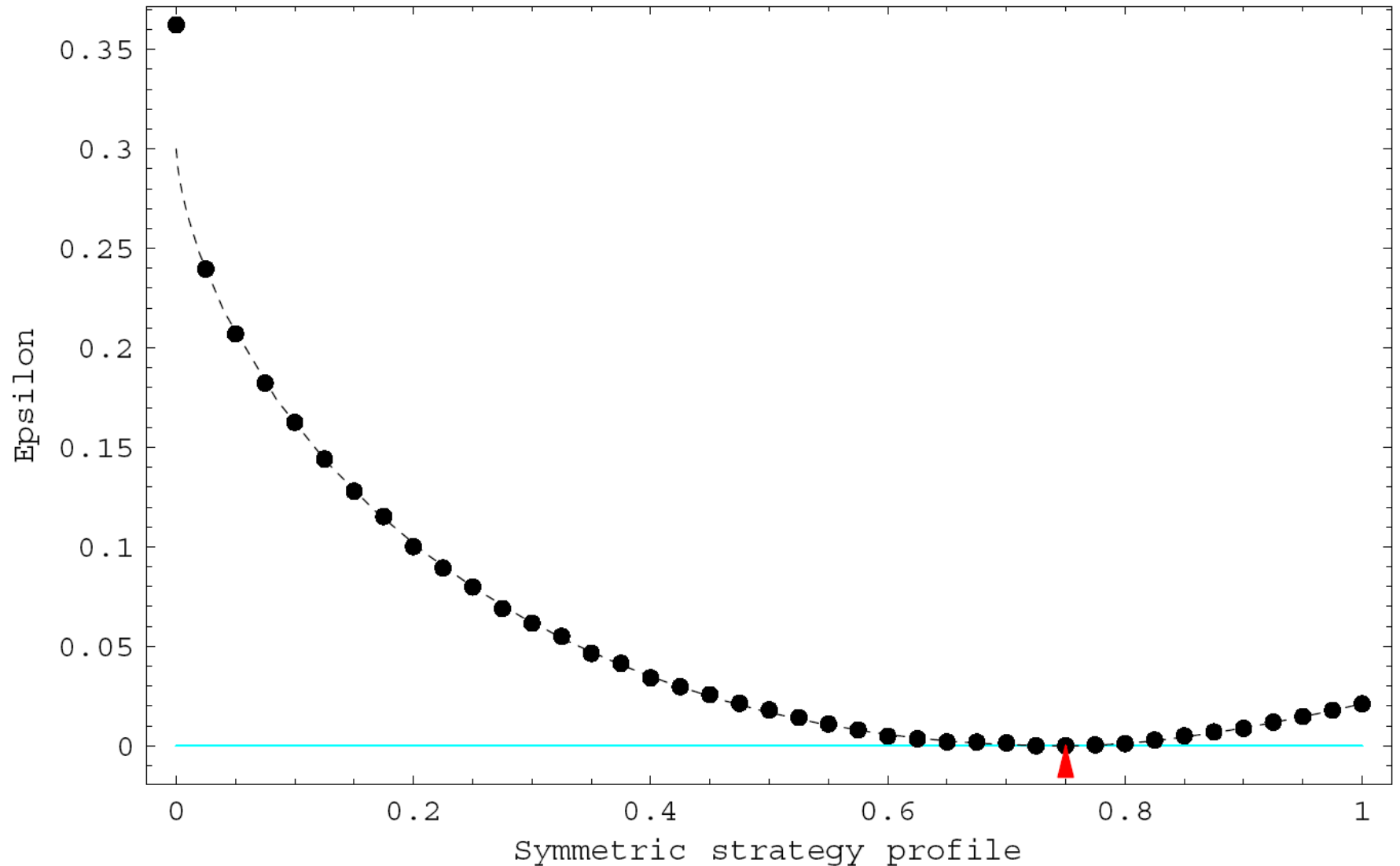
Game Simulation: FPSB4



Game Simulation: FPSB4



Game Simulation: FPSB4



Control Variates

- Variance reduction by *adjusting for luck*
- FPSB: higher valuation means higher expected payoff
- Suppose $g(t)$ estimates payoffs; adjust sampled payoffs by subtracting $g(t) - E[g(t)]$

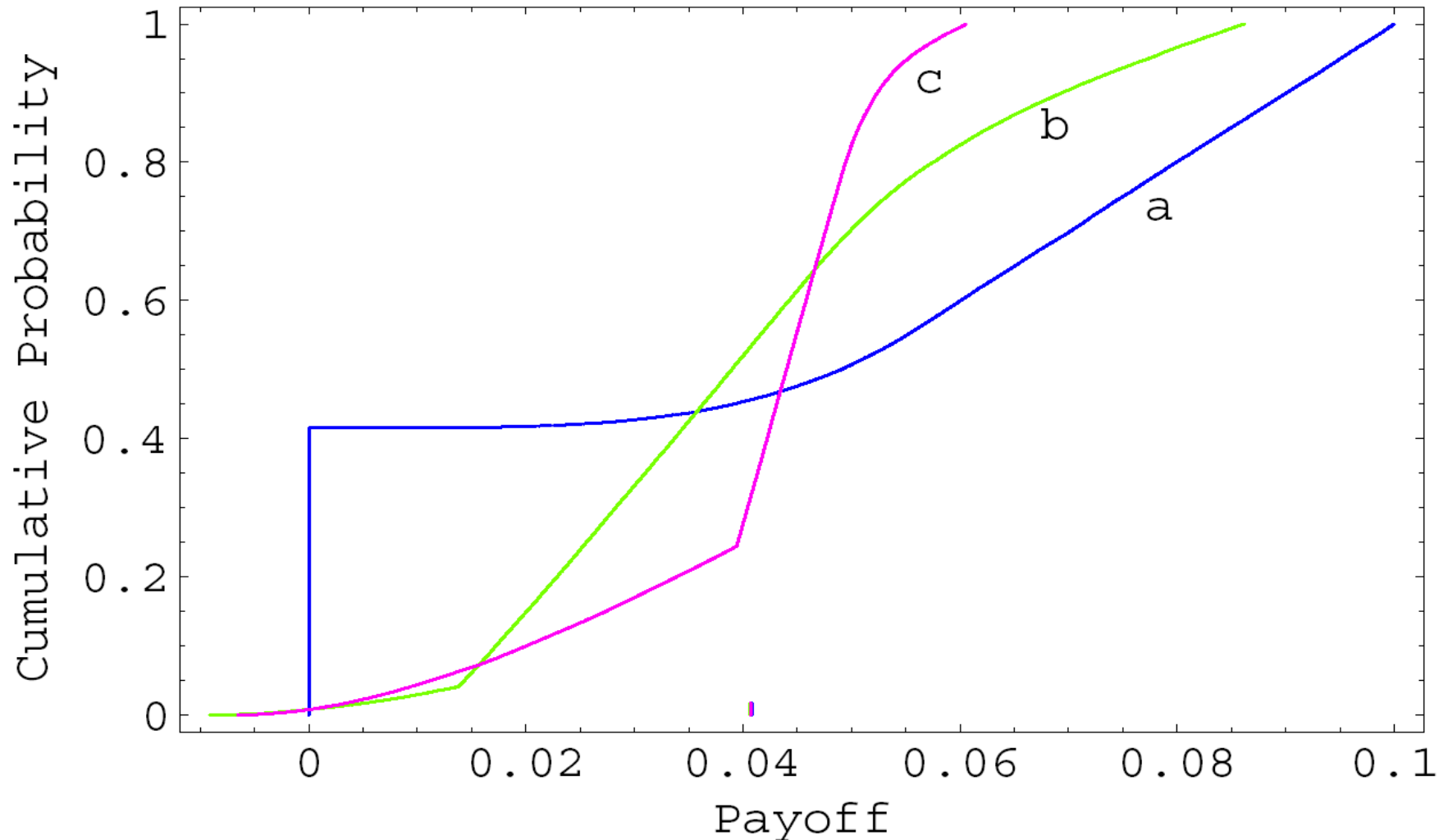
$$g(t_i) = \frac{(1 - k_i)k_i^{n-1}t_i^n}{\prod_{j \neq i} k_j},$$

$$E[g(t_i)] = \frac{(1 - k_i)k_i^{n-1}}{(n + 1) \prod_{j \neq i} k_j}.$$

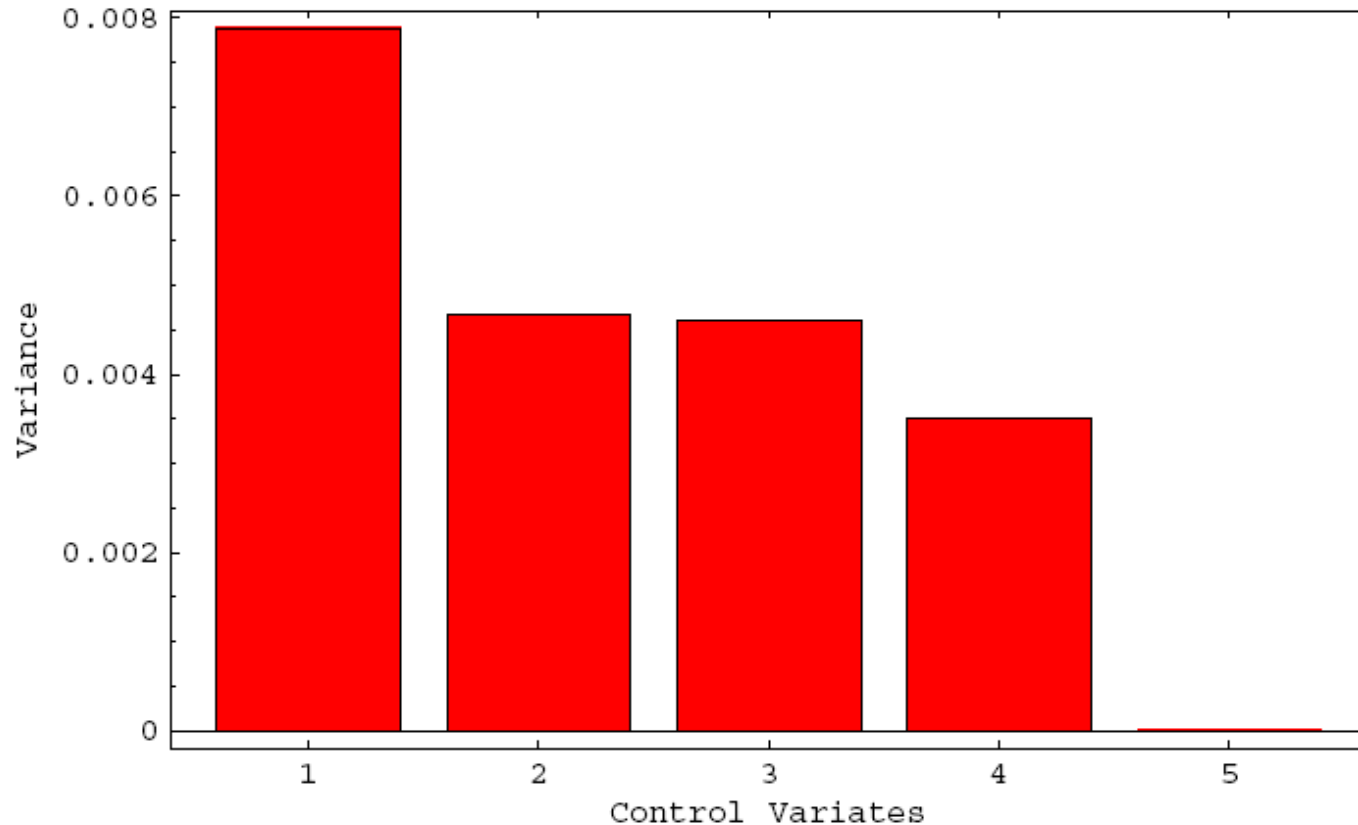
Control Variates: Estimating $g(t)$

- Bootstrapping a $g()$ – learning a function from types to payoffs from empirical samples
- Semi-automated approach: pick summary statistics and perform linear regression
 - $g(t) = B x(t) + A$
- Unbiased as long as the regression parameters are estimated from a distinct data set (exogenously determined $g()$)

Unadjusted and adjusted Payoffs in FPSB



Average Variance Reduction



- 1. Unadjusted (0)
- 2. Linear regression, profile non-specific
- 3. Linear regression for this profile
- 4. Analytically determined $g()$
- 5. $g() =$ exact expectation (sanity check)

Other Variance Reduction Techniques

- Quasi-random Sampling
- Importance Sampling
- Stratified Sampling
- Combined and Adaptive techniques
- (Application requires special handling of randomness from Nature)

Player Reduction

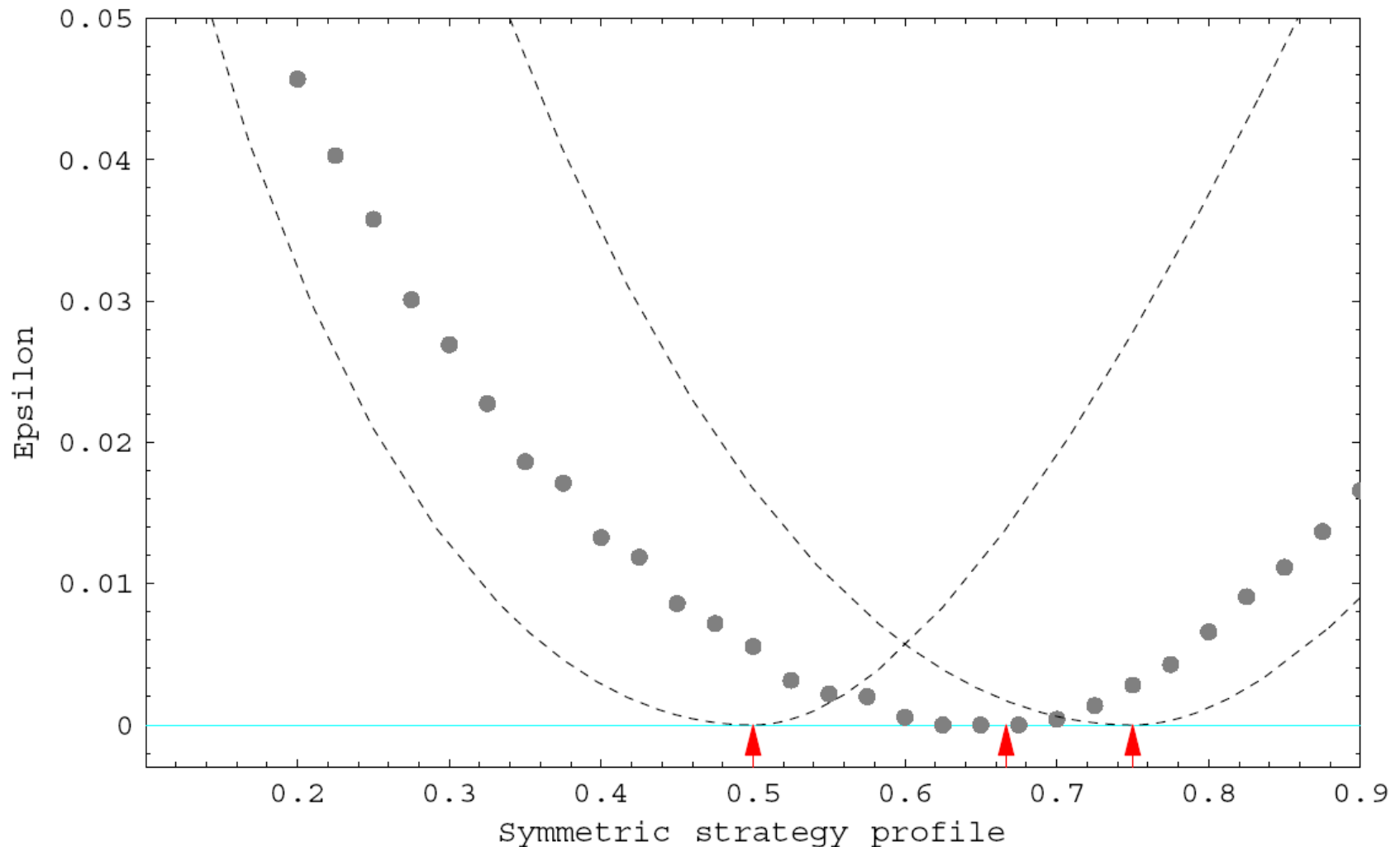
- Definition: $\Gamma \downarrow_p$

$$\hat{u}_i(s_1, \dots, s_p) = u_{q \cdot i}(\underbrace{s_1, \dots}_{q}, \underbrace{s_2, \dots}_{q}, \dots, \underbrace{s_p, \dots}_{q})$$

- Theorem: unique symmetric equilibrium of $\text{FPSB}n \downarrow_p$ is

$$\frac{n(p-1)}{p + n(p-1)}$$

Epsilons for Symmetric Profiles in FPSB2, FPSB4₂, FPSB4



Theoretical Results for FPSB

- Theorem: For all $n > p \geq 1$

$$eq(\text{FPSB}p) < eq(\text{FPSB}n \downarrow_p) < eq(\text{FPSB}n) \quad \text{and}$$

$$0 = \varepsilon(eq(\text{FPSB}n)) < \varepsilon(eq(\text{FPSB}n \downarrow_p)) < \varepsilon(eq(\text{FPSB}p))$$

(reducing to p players always outperforms the p player version)

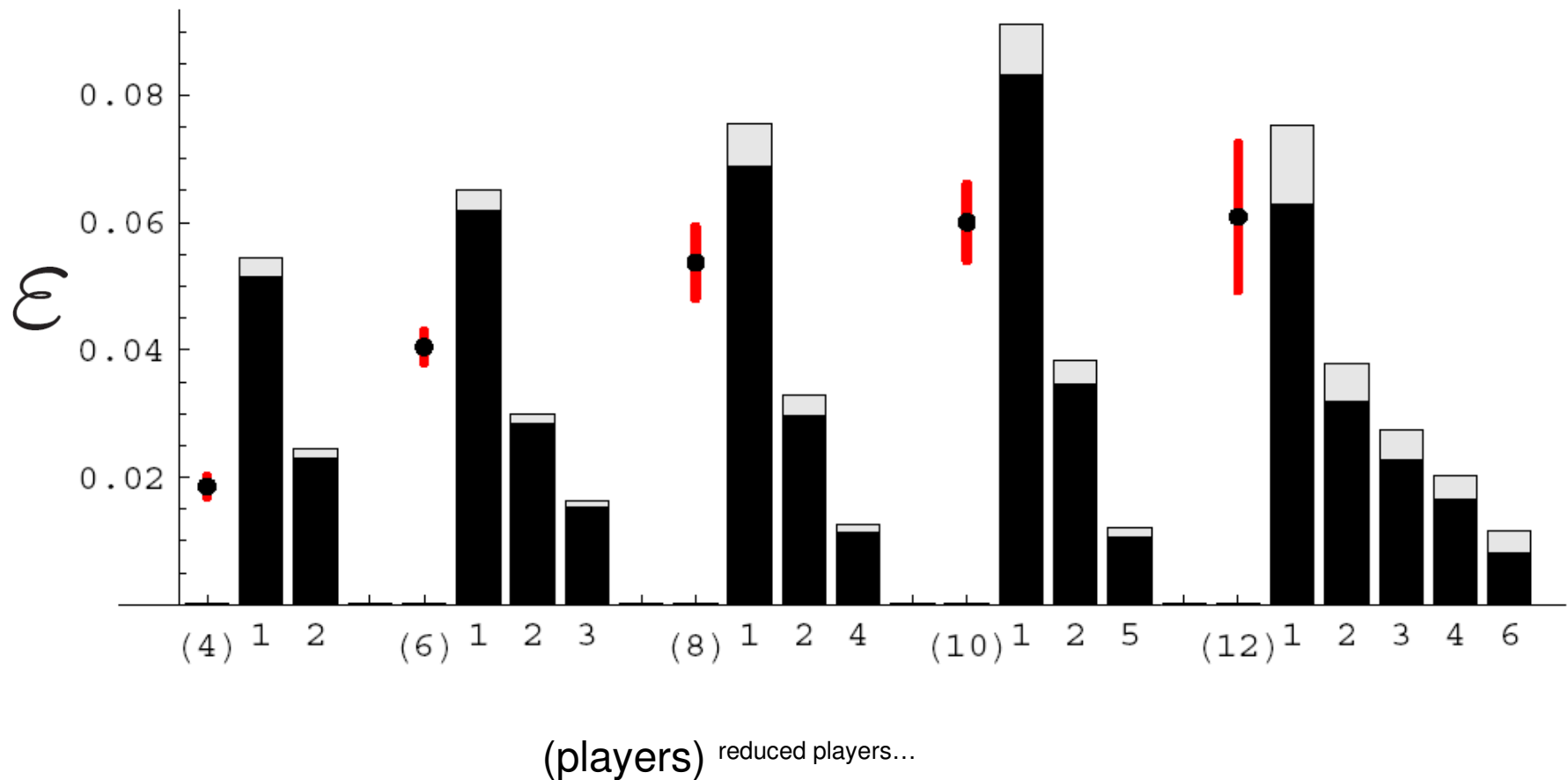
- Theorem: For all $n > p > q \geq 1$

$$eq(\text{FPSB}n \downarrow_q) < eq(\text{FPSB}n \downarrow_p) < eq(\text{FPSB}n) \quad \text{and}$$

$$0 = \varepsilon(eq(\text{FPSB}n)) < \varepsilon(eq(\text{FPSB}n \downarrow_p)) < \varepsilon(eq(\text{FPSB}n \downarrow_q))$$

(solution quality degrades monotonically with more severe player reduction)

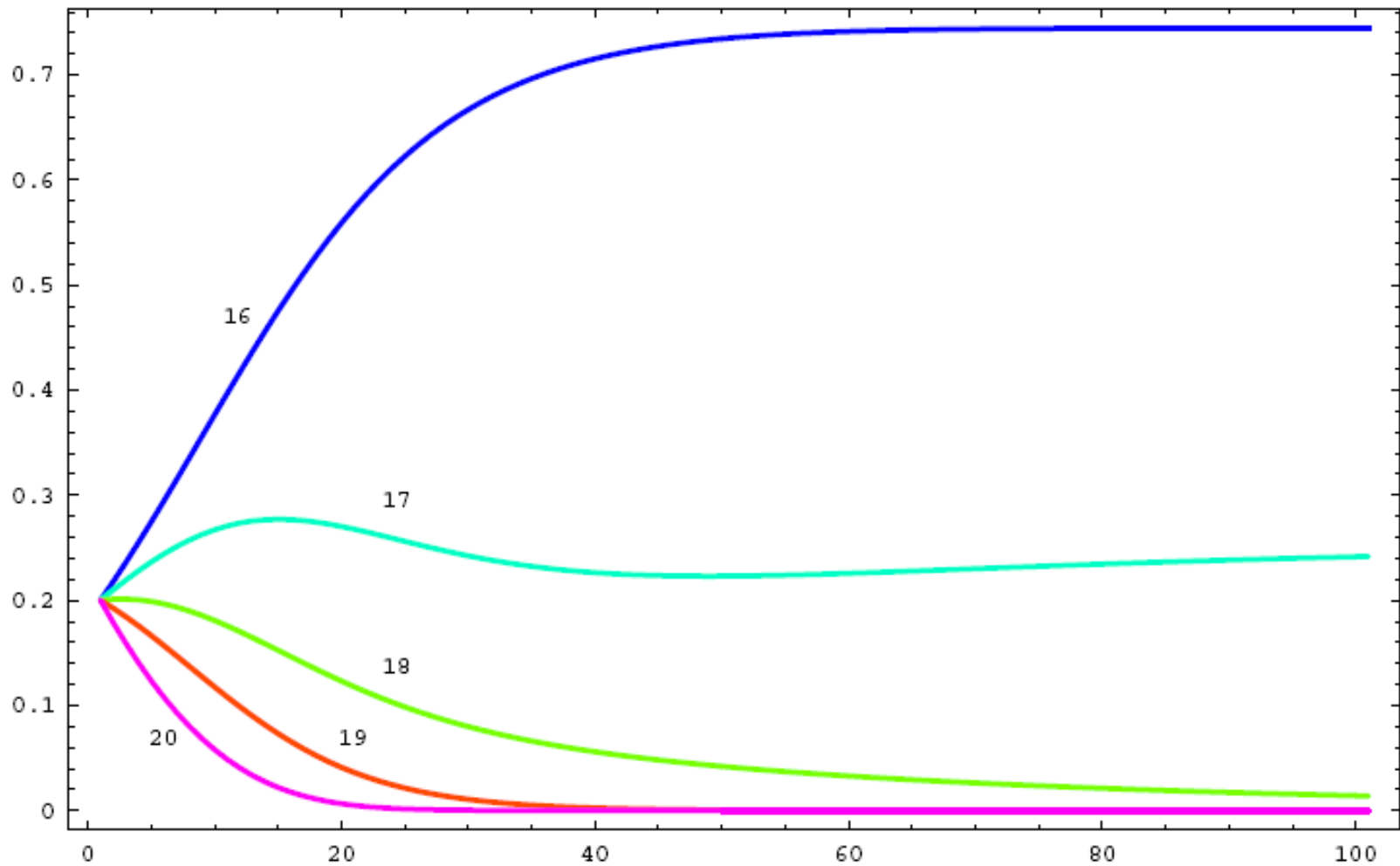
Local-Effect Games



Analyzing Empirical Games

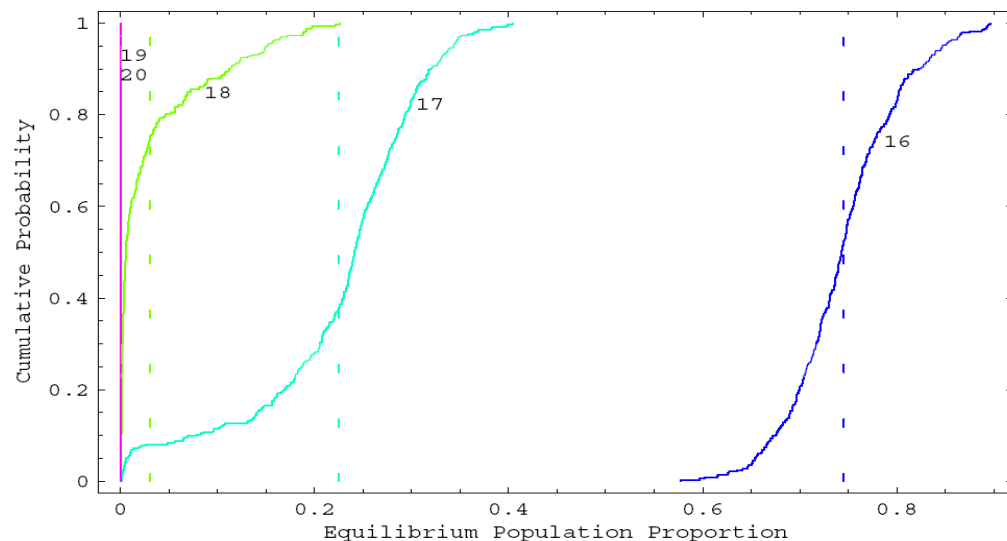
- Gambit
- Replicator Dynamics
- Function Minimization (Amoeba)

Example of Replicator Dynamics

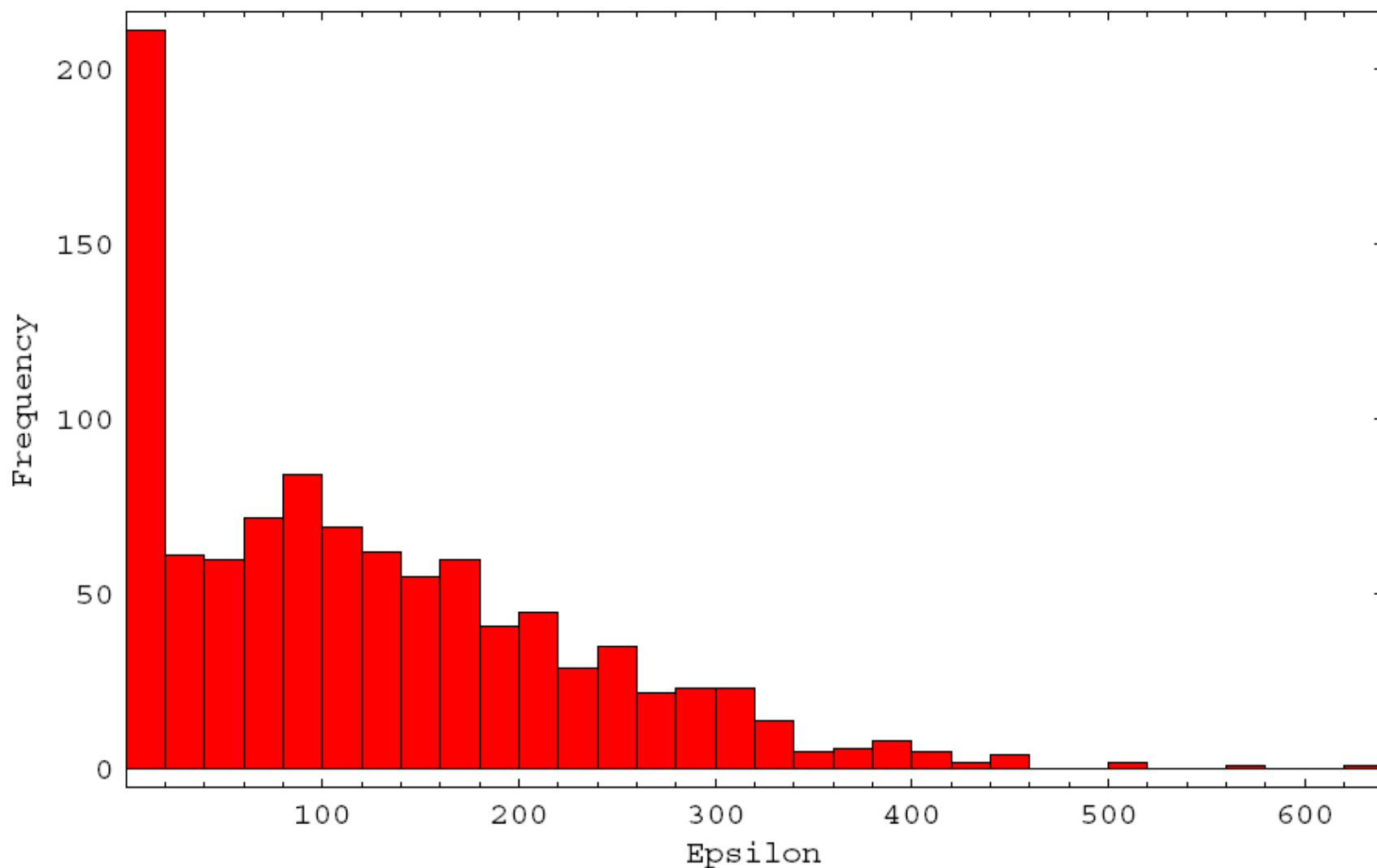


Sensitivity Analysis

- Distributions over payoff matrices
- Distributions over functions of the payoff matrix
 - Nash equilibrium
 - Epsilon for a Nash *candidate*
- Confidence bounds on mixture probabilities
- Confidence bounds on epsilon



Sensitivity Analysis

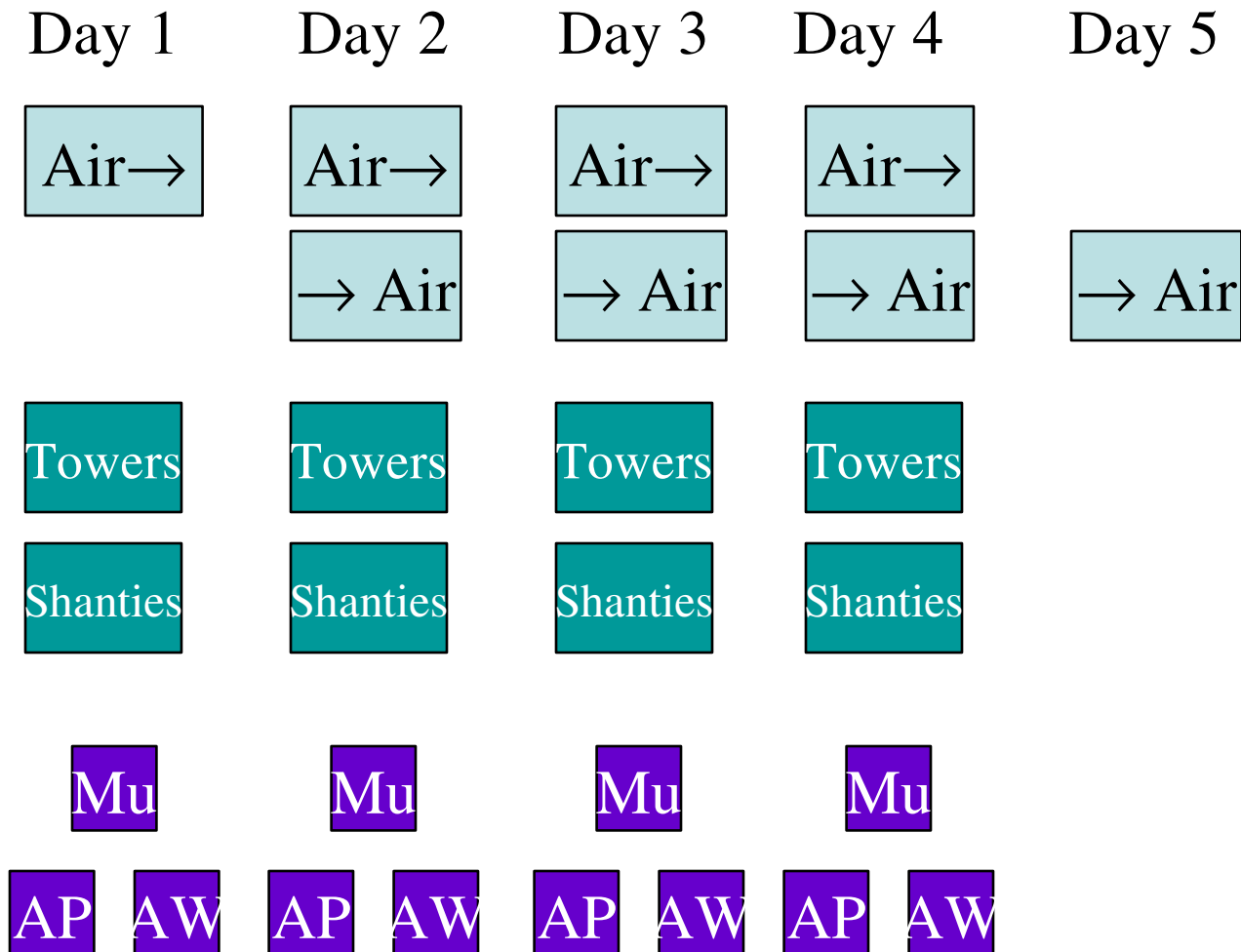


Conclusion:

Taming Monster Games

- Restrict strategy space
- Reduce number of players
- Simulate game outcomes
 - Adjust type distributions or adjust sampled payoffs with control variables to reduce variance
- Analyze empirical game
- Assess solutions wrt underlying game

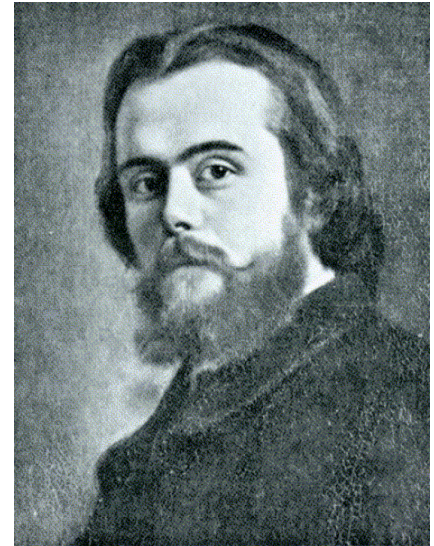
(Simultaneous Auctions in) TAC Travel



“Walverine”



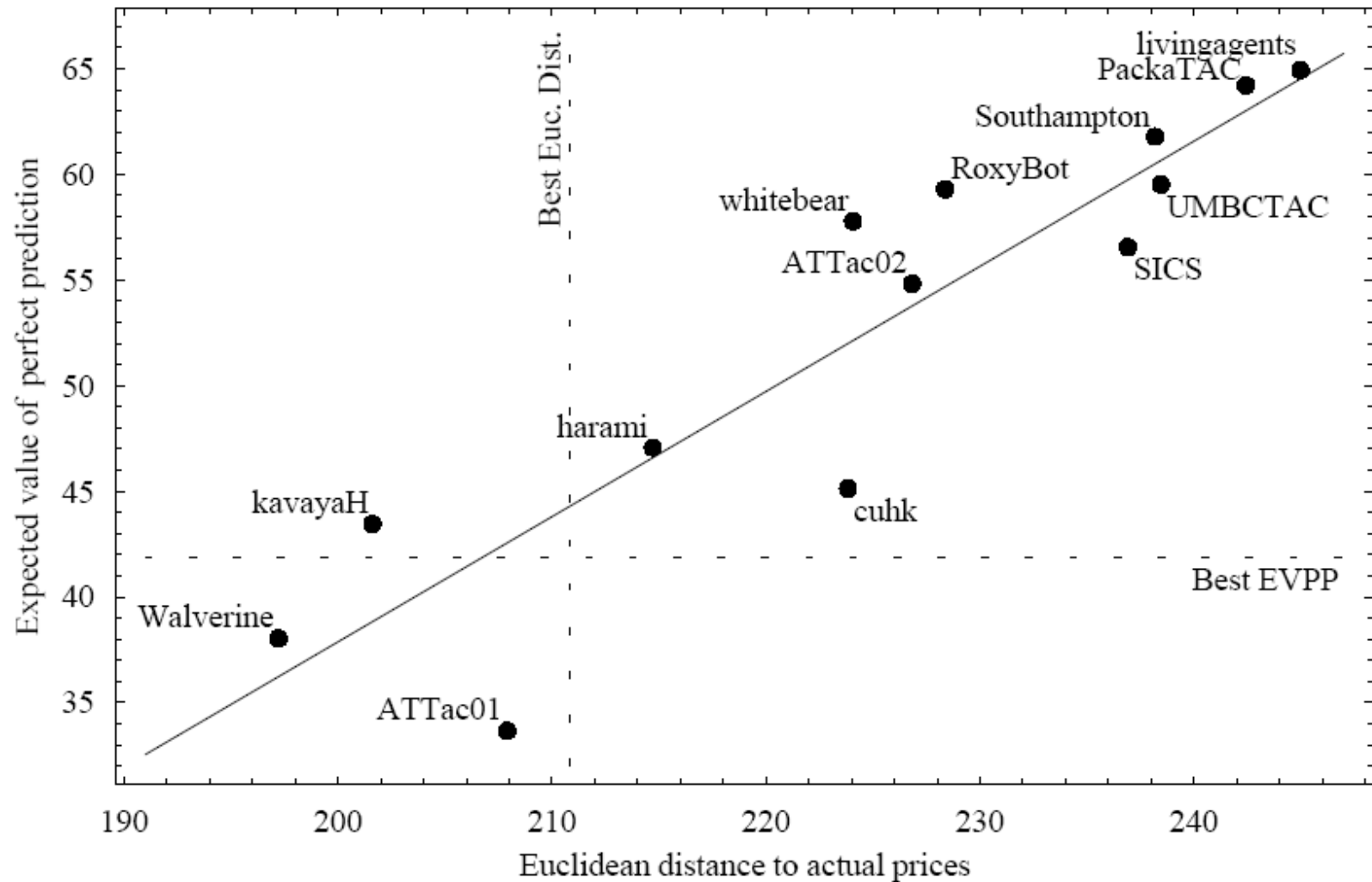
wolverine



Léon Walras

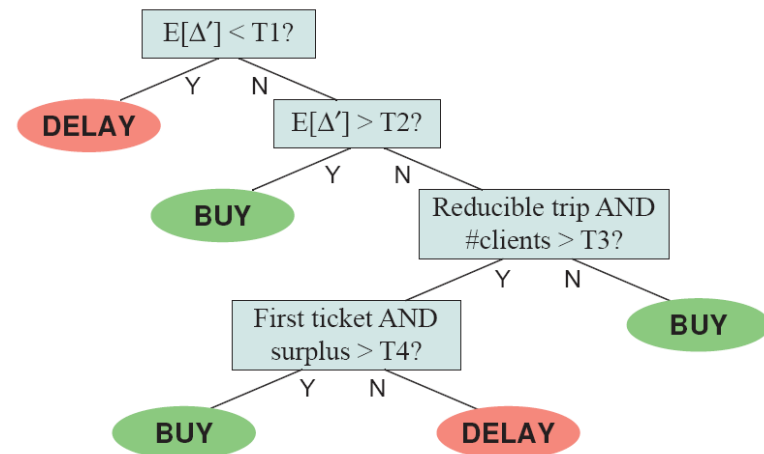


TAC Price Prediction



Parameterized TAC Agent

- Walrasian Price Predictor with...
- Variations on hotel bid shading
- Different entertainment trading strategies
- Parameterized decision process for flight purchase timing

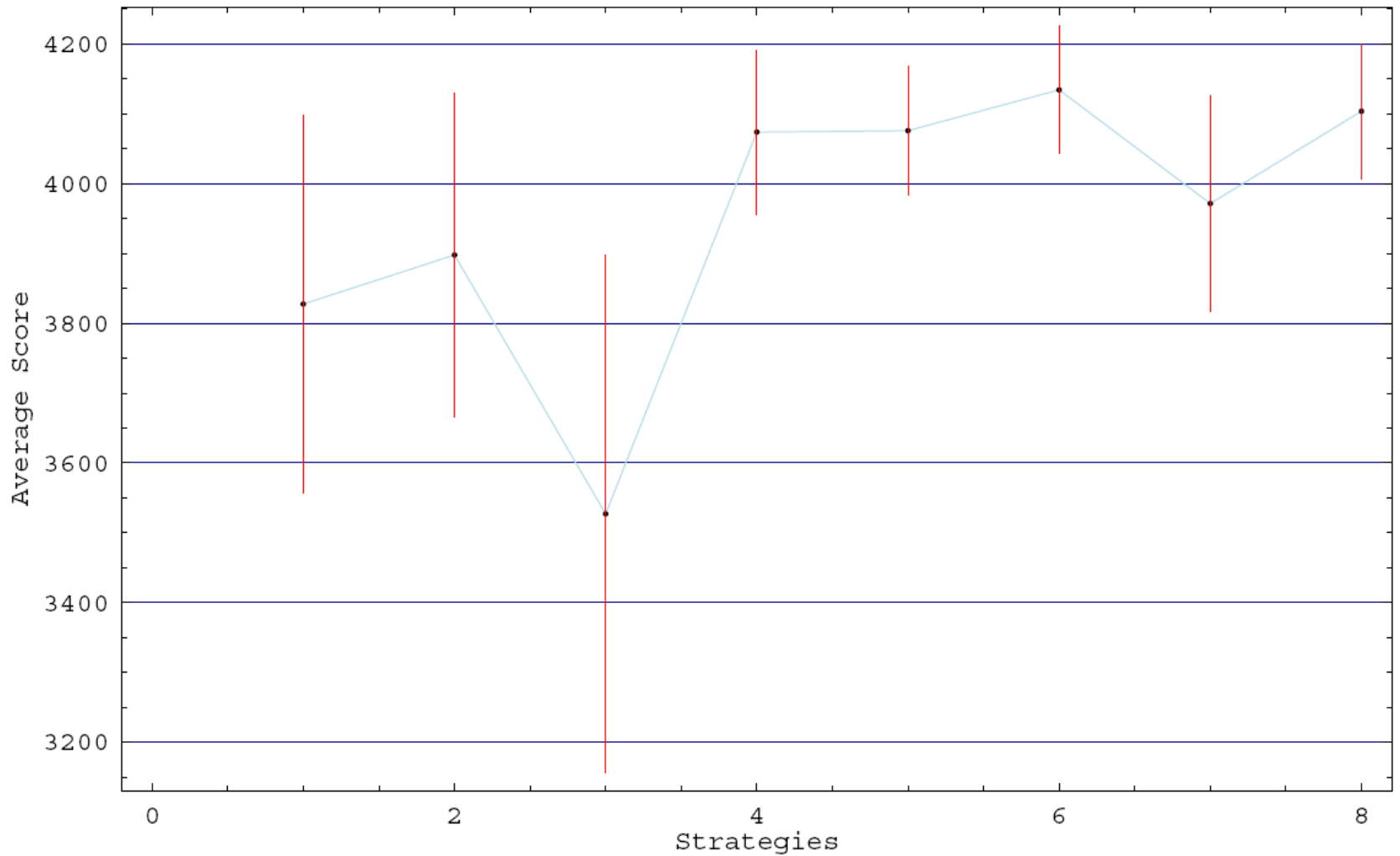


Searching for Walverine (among 40 candidate strategies)

p	Profiles			Samples/Profile	
	total	evaluated	%	min	mean
8	> 314M	2114	0	12	22.3
4	123,410	2114	1.7	12	22.3
2	820	586	71.5	15	31.7
1	40	40	100	25	86.5

Profiles Evaluated in Reduced TAC Games

Performance of 8 Wolverine Variants



Walverine in Third...



WELCOME TO TRADING AGENT COMPETITION

Competitive Benchmarking for The Trading Agent Community

[LOGIN](#) | [SEARCH](#) | [FAQ](#) | [LEGAL NOTICES](#)

GENERAL

[Home](#)
[About TAC](#)
[News](#)
[Press](#)
[Calendar](#)
[Contact Info](#)

JOIN NEWS LIST

TAC 2005

[Info & Call](#)
[Participants](#)
[TADA Workshop](#)
[Results](#)

PREVIOUS RESULTS

[TAC 2004](#)
[TAC 2003](#)
[TAC 2002](#)

TAC COMMUNITY

[Research Groups](#)
[Educating using TAC](#)
[TAC Policies](#)

[TAC 2005 Finals](#) | [TAC 2005 Semi-Finals](#) | [TAC 2005 Seeding](#) | [TAC 2005 Qualifying](#)

Score tables for TAC 2005 Finals - 3rd August

TAC Classic

Position	Agent	Average Score	Games Played	Zero Games
1	Mertacor (tac1 , tac2)	4126.49	80	0
2	whitebear05 (tac1 , tac2)	4105.68	80	0
3	Walverine (tac1 , tac2)	4058.90	80	2
4	Dolphin (tac1 , tac2)	4022.79	80	0
5	SICS02 (tac1 , tac2)	3972.29	80	0
6	LearnAgents (tac1 , tac2)	3899.24	80	0
7	e-Agent (tac1 , tac2)	3451.25	80	0
8	RoxyBot (tac1 , tac2)	3167.64	80	10

The scores have been combined from the TAC 2005 finals at [tac1.sics.se](#) and [tac2.sics.se](#).

TAC SCM

Walverine Rules (p=.17)

Recalculated Scores for TAC 2005 Finals

Position	Agent	Average Score	Games Played	Zero Games
1	Walverine (tac1 , tac2)	4157.10	58	0
2	RoxyBot (tac1 , tac2)	4066.94	58	0
3	Mertacor (tac1 , tac2)	4063.17	58	0
4	whitebear05 (tac1 , tac2)	4001.89	58	0
5	Dolphin (tac1 , tac2)	3993.31	58	0
6	SICS02 (tac1 , tac2)	3904.70	58	0
7	LearnAgents (tac1 , tac2)	3785.36	58	0
8	e-Agent (tac1 , tac2)	3369.99	58	0

The scores have been combined from the TAC 2005 finals at [tac1.sics.se](#) (games 30563-30591) and [tac2.sics.se](#) (games 12108-12136).

This is a recalculation of the TAC 2005 final scores where some problem games have been removed (RoxyBot was running two agents on [tac1.sics.se](#) and no agent at [tac2.sics.se](#) during the first games).

Strategy Generation Summary

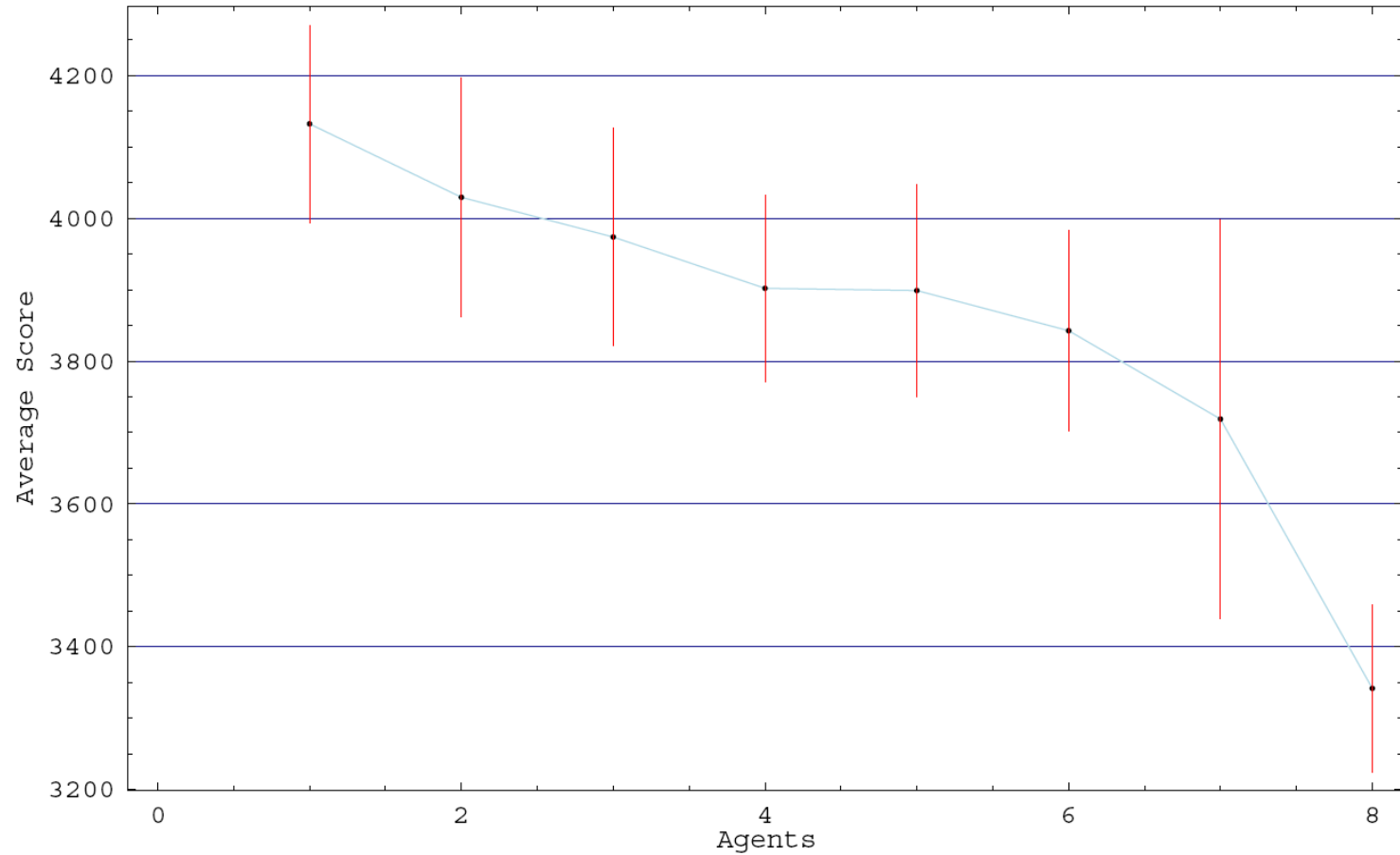
- First algorithm to compute best-response strategies in a broad class of infinite games of incomplete information
- Empirical game methodology for applying game-theoretic analysis to much larger games than previously possible
- Theoretical and experimental evidence of the efficacy of our methodology
- A price-prediction approach to strategy generation in simultaneous auctions for complementary goods
- Application of the above methods to find good strategies in complex games (TAC and SAA)
- **Future Work:** new domains (multiattribute auctions, sponsored search auctions), computational mechanism design

Backup Slides

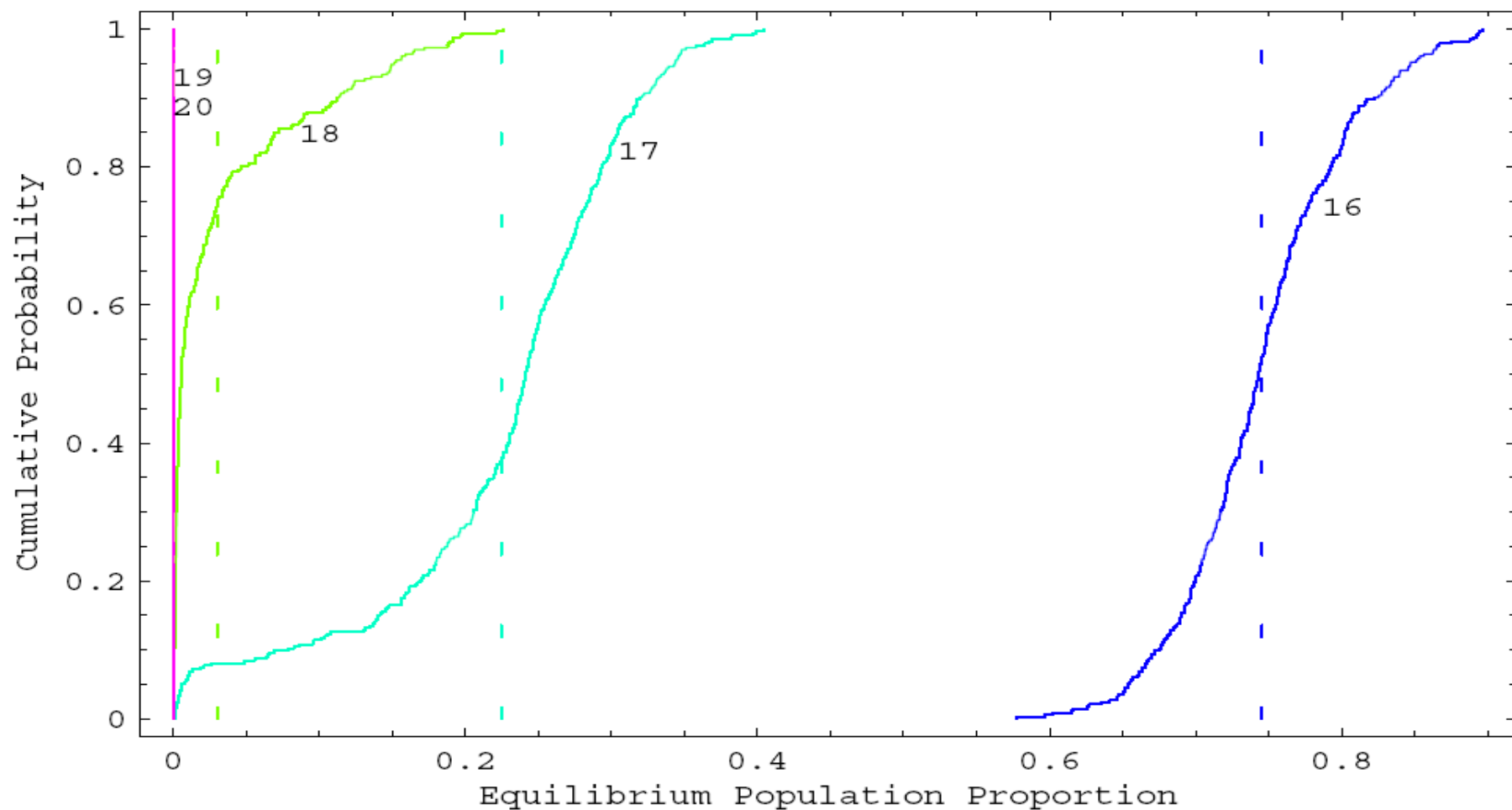
Strategy Metrics

Strategy	Count	Max	Sum
17	24	0.499	5.39
4	19	0.729	4.86
21	17	0.501	4.68
16	18	0.892	3.77
23	14	0.542	3.34
6	16	0.699	3.25
9	16	0.367	3.09
5	23	0.247	2.63
24	17	0.232	2.34
35	15	0.641	2.04
40	20	0.180	1.95
3	8	0.401	1.70
34	7	0.307	1.05
7	6	0.099	0.37
38	5	0.091	0.35
39	3	0.126	0.18

Comparison of TAC-05 Scores



Sensitivity Analysis 1



Price Prediction for Complementary Goods in Simultaneous Auctions

- Point vs. distribution prediction
- Self-confirming prediction
- Walrasian price equilibrium prediction
 - Prices such that supply meets demand
- Both can be found (at least approximately) by an iterative process

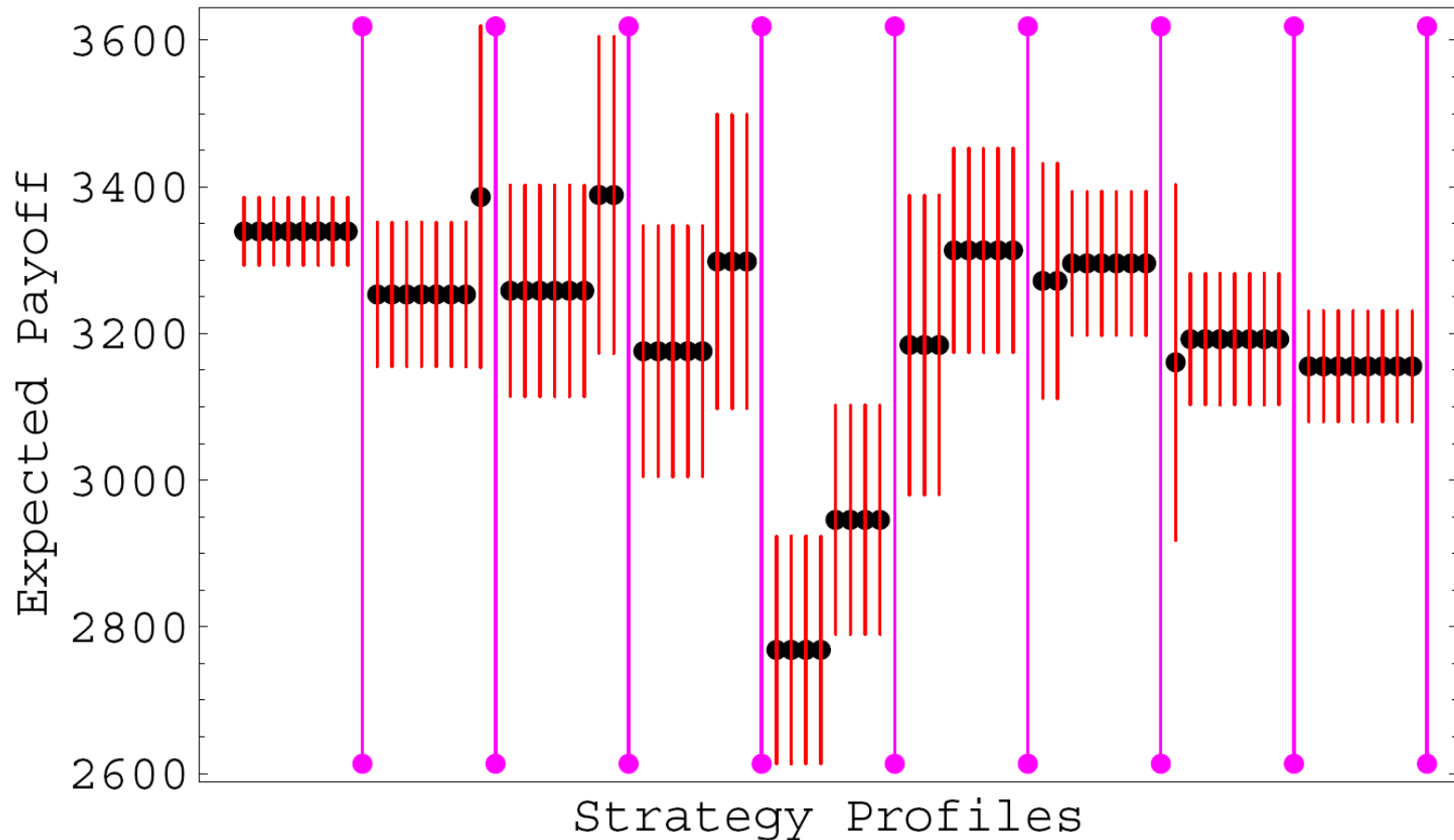
Performance of Self-Confirming Prediction in SAA Games

$\text{SAA}_{\lambda \sim *}(m, n)$	$\varepsilon\% \left(\text{PP}(F^{\text{SC}}) \right)$	$\overline{\varepsilon\%}$	$\text{Pr}(\varepsilon = 0)$	$\text{Pr} \left(\text{PP}(F^{\text{SC}}) \right)$
$E(3, 3)$	0	0	1.00	1.00
$E(3, 5)$	0	.09	.600	.996
$E(3, 8)$.83	.85	0	—
$E(5, 3)$	0	0	1.00	.999
$E(5, 5)$	0	.01	.900	.998
$E(5, 8)$.60	.64	0	—
$E(7, 3)$	0	.06	.667	.992
$E(7, 6)$.04	.10	.567	.549
$U(3, 3)$	1.24	1.26	0	.725
$U(3, 5)$	0	0	1.00	1.00
$U(3, 8)$.56	.53	0	—
$U(5, 3)$	1.35	1.35	0	.809
$U(5, 8)$	1.59	1.62	0	—
$U(7, 3)$.81	.84	0	.942
$U(7, 6)$.52	.52	0	.929
$U(7, 8)$	4.98	4.94	0	—

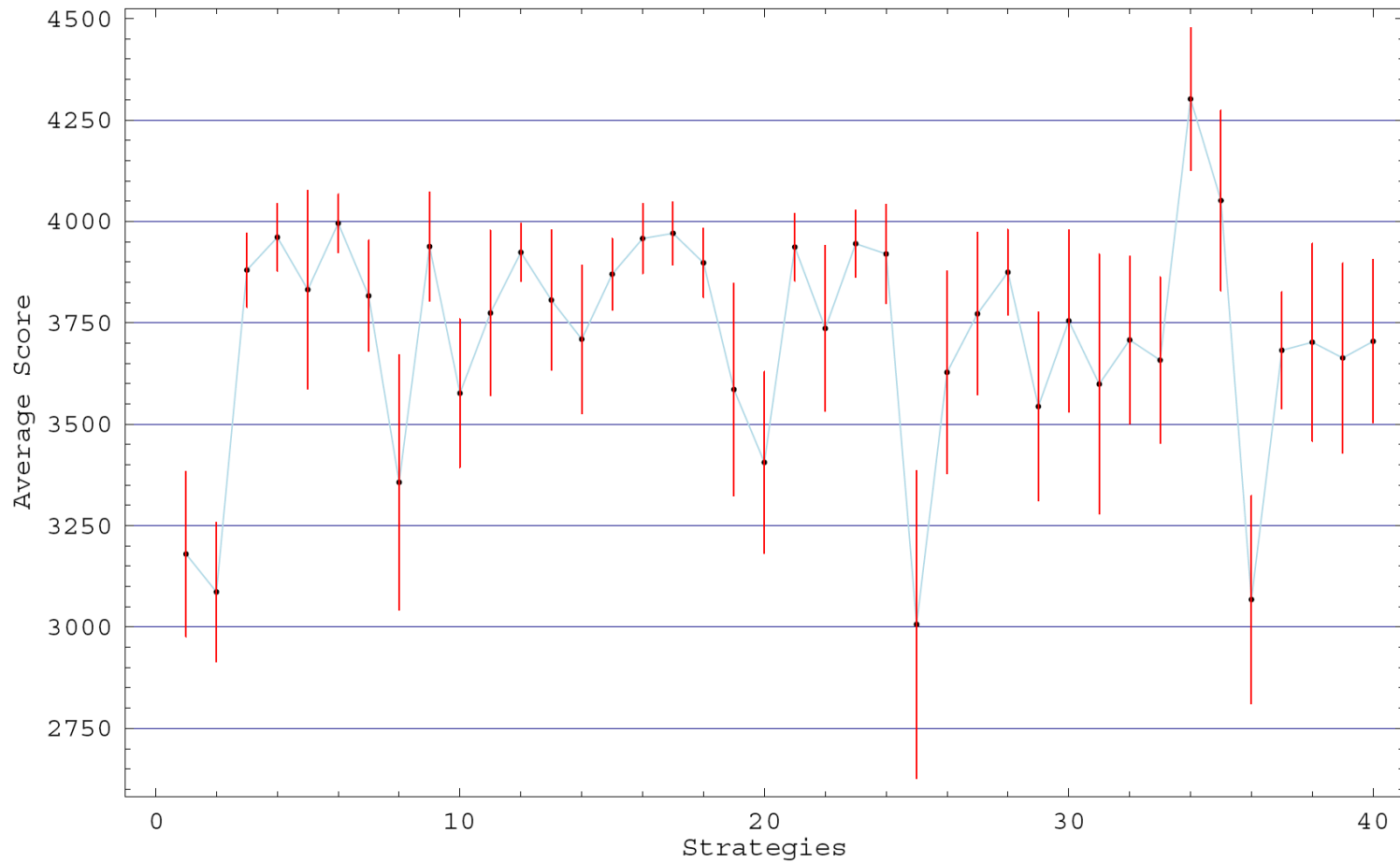
The Trading Agent Competition

- 12 minute games, 8 agents competing per game
- Agents perform as travel agents, purchasing travel goods for clients at auction
- Each travel agent given clients requests, defining objective function. Net value is objective minus expenditure
- Assemble trip for each client, comprising flight, hotel, and entertainment
- Goods are interdependent, each presents interesting issues

Shading vs. Non-Shading Walverine



Symmetric Strategies in TAC 1



Prisoners' Dilemma

	17 (D)	34 (C)
17 (D)	3971	4377
34 (C)	3907	4302

Why “Strategy Generation”? (SG)

- Infinite Games: SG is meant in the sense of a game solver (generating equilibrium strategies from a game description)
- Monster Games: SG refers to the process of generating a set of candidate strategies as well as choosing among them
- Price Prediction: families of strategies from which we generate prediction specific strategies
- TAC/SAA: applying empirical game methodology to establish good strategies in two market games

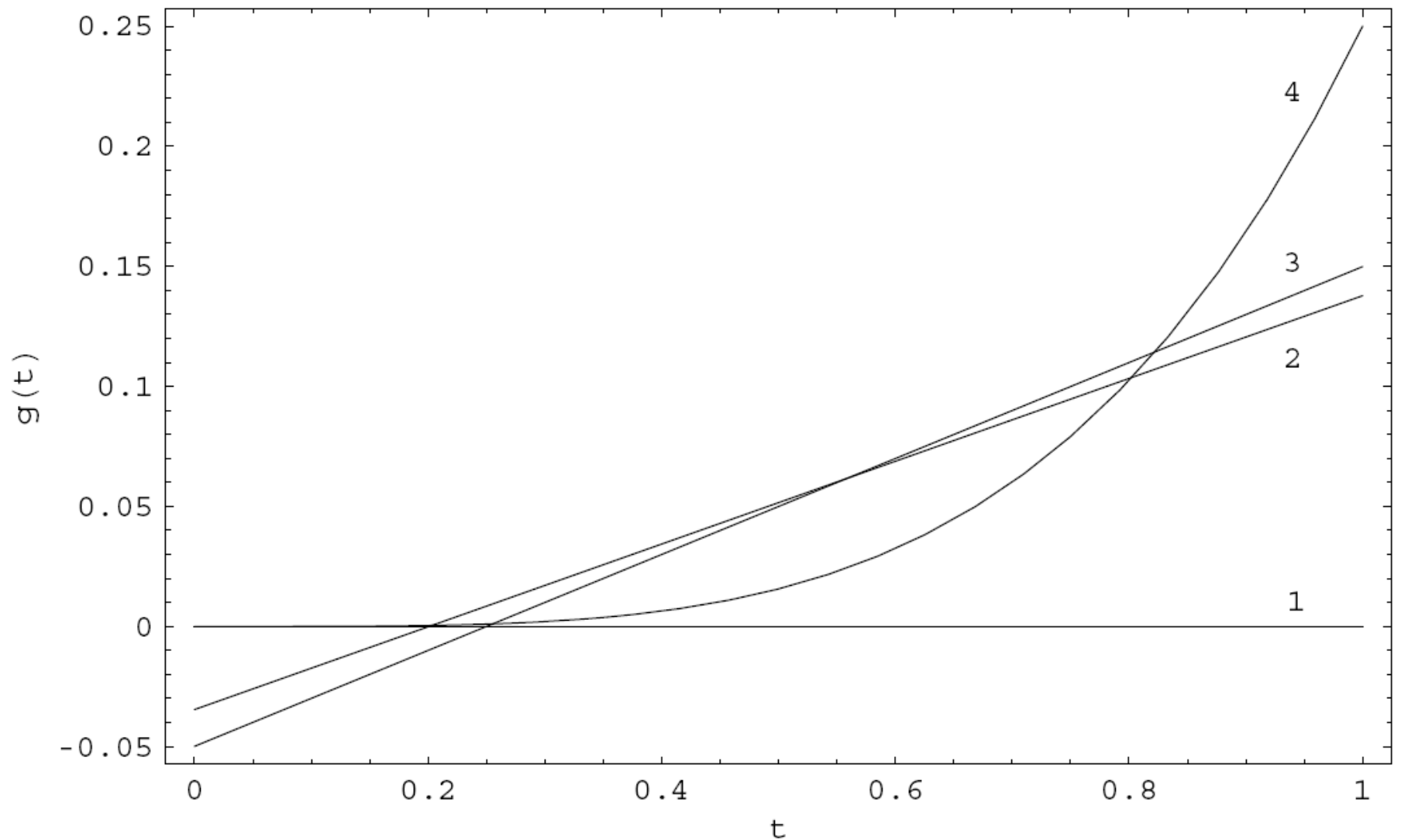
TAC-05 (if not for meddling kids)

Agent	Raw Score	Adjusted Score	95% C.I.
Walverine	4157.1	4132.42	± 138.4
RoxyBot	4066.94	4029.53	± 167.3
Mertacor	4063.17	3973.9	± 152.3
whitebear05	4001.89	3902.01	± 130.4
Dolphin	3993.31	3898.95	± 148.5
SICS02	3904.7	3842.61	± 140.6
LearnAgents	3785.36	3718.81	± 280.0
e-Agent	3369.99	3341.52	± 117.2

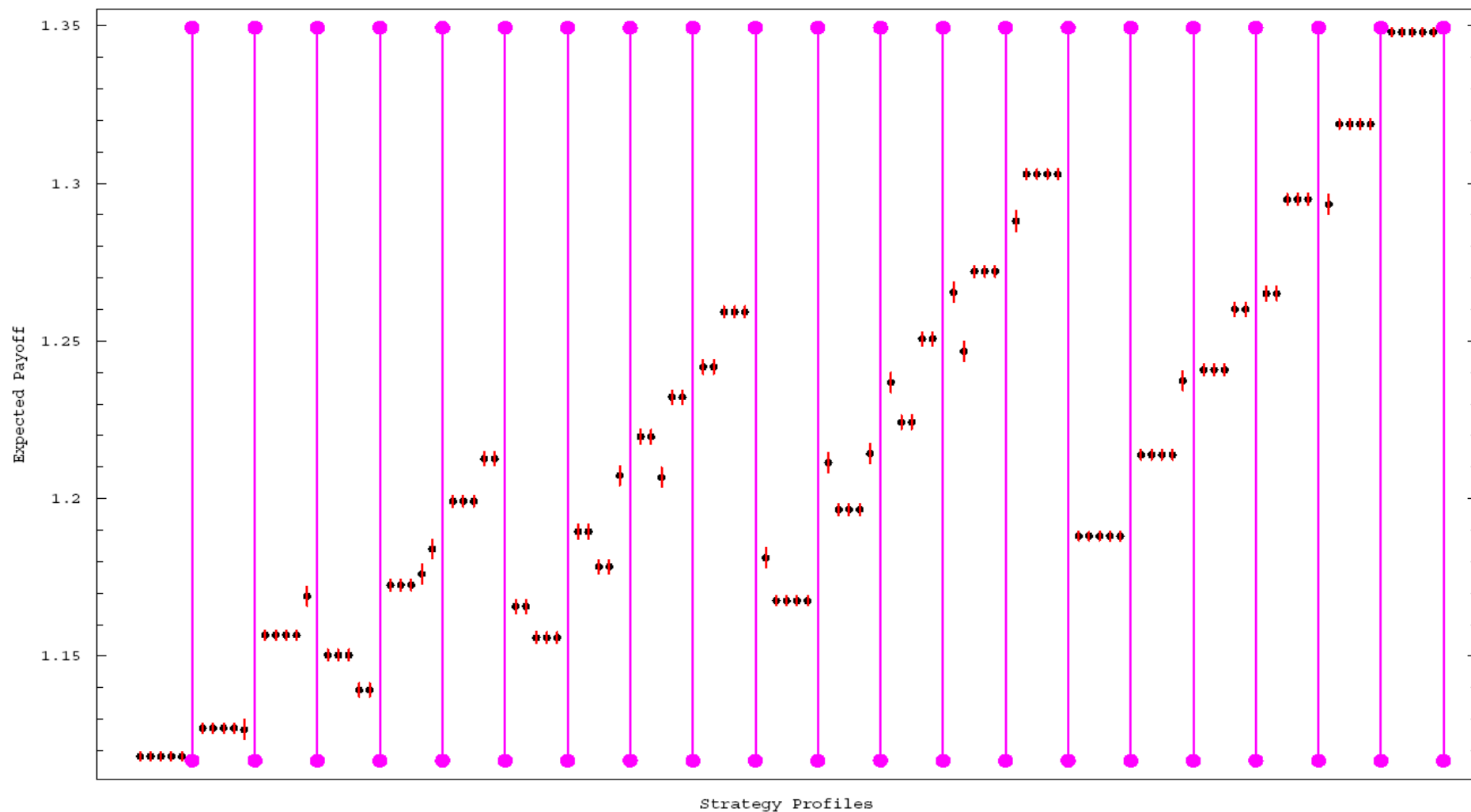
Price Prediction Can Help

Agent	$\{1\}$	$\{2\}$	$\{3\}$	$\{1, 2\}$	$\{1, 3\}$	$\{2, 3\}$	$\{1, 2, 3\}$
1	0	0	0	0	0	0	15
2	8	6	5	8	8	6	8
3	10	8	6	10	10	8	10

Control Variates for FPSB4



Empirical Payoff Matrix



Finite Game Approximations

- Finite game solvers:
 - Gambit
 - Gala
 - Gametracer
- Why not discretize?
 - Introduces qualitative differences
 - Computationally intractable

Price Prediction Strategies for Market-Based Scheduling

Jeffrey K. MacKie-Mason

Anna Osepayshvili

Daniel M. Reeves

Michael P. Wellman

University of Michigan

Factory Scheduling Example

Agent 1

value = \$10

length = 2hr.

deadline = 13:00

Agent 3

value = \$6

length = 1hr.

deadline = 12:00

Factory

9:00

10:00

11:00

12:00

13:00

14:00

15:00

16:00

Agent 2

value = \$16

length = 2hr.

deadline = 12:00

Agent 4

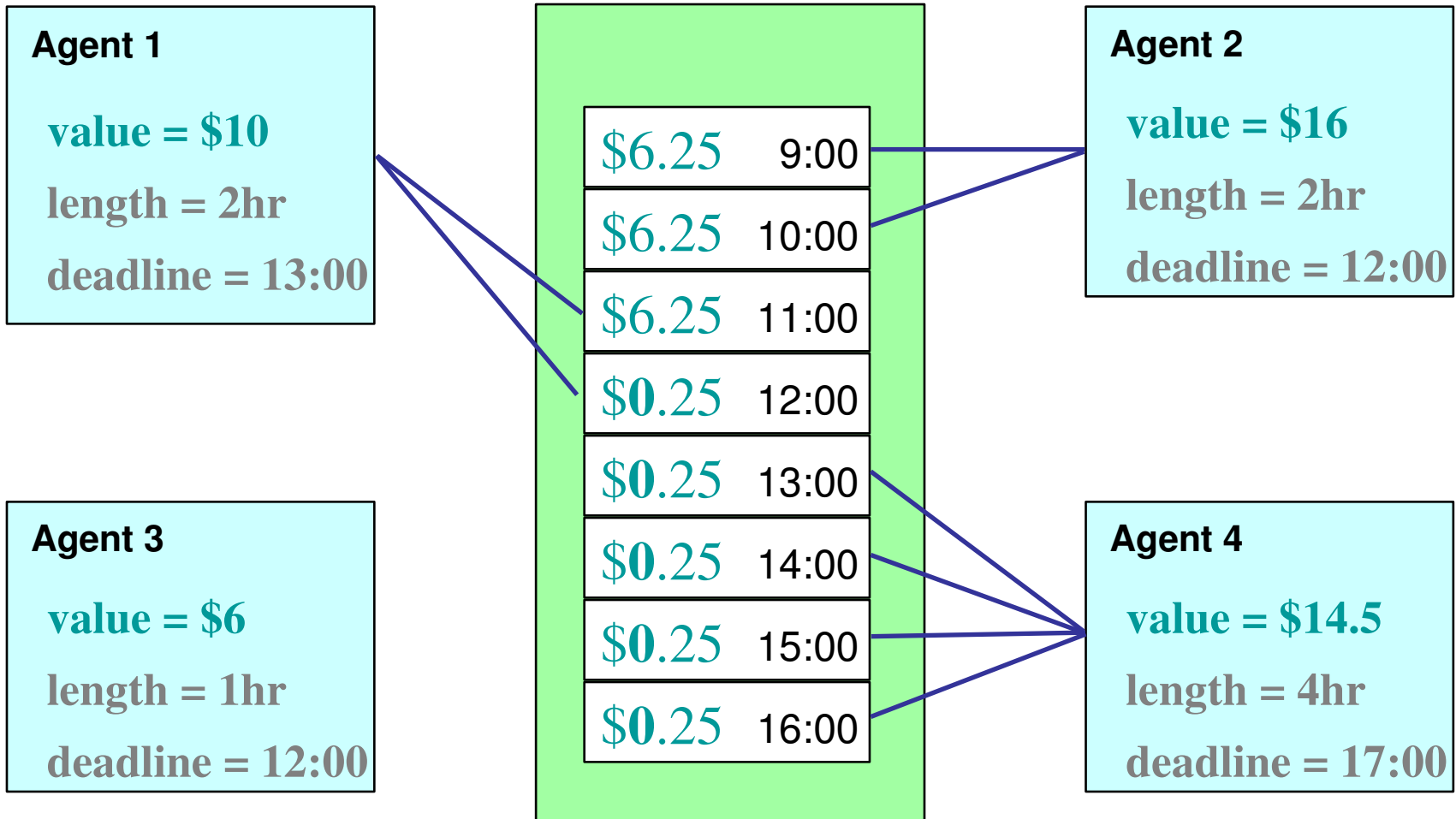
value = \$14.5

length = 4hr.

deadline = 17:00

Schedule Price Equilibrium

Factory



The Market Mechanism

- Agent Preferences
 - Job length
 - Deadline values
- **Simultaneous Ascending Auctions**
 - One auction per time slot
 - Price quotes announced after each round
 - Auctions clear when all are quiescent

Exposure Problem


- Balance benefit of acquiring enough slots with risk of buying unusable slots
- Price prediction can mitigate the exposure problem
- Knowing the eventual price of a slot means you can avoid committing to it

Name	Job Length (λ)	$v(1)$	$v(2)$	$v(3)$
Agent 1	3	—	—	15
Agent 2	1	8	6	4
Agent 3	1	10	8	6

Straightforward Bidding (SB)

No attempt to anticipate other agents' strategies:

- Perceived prices
- Best bundle
 - maximizes surplus at perceived prices
 - assumes it will win the whole bundle

Slots	1	2	3	
Current prices (CP)	\$10	\$9	\$5	
Slots the agent is winning	yes	---	---	
Perceived prices 	\$10	\$10	\$6	
Job length	2 slots			Surplus = Value - Cost
Deadline Values	---	\$35	\$25	
Bundle {1,2}	yes	yes	---	
Bundle {1,3}	yes	---	yes	
Bundle {2,3}	---	yes	yes	
				35 - 20 = \$10
				25 - 16 = \$9
				25 - 16 = \$9

Modification Of SB

Estimate final prices:

- Perceived prices
- Information on market prices
- Best bundle
 - maximizes surplus at estimated final prices
 - assumes it will win the whole bundle

Slots	1	2	3	
Current prices (CP)	\$10	\$9	\$5	
Slots the agent is winning	yes	---	---	
Perceived prices	\$10	\$10	\$6	
Vector of predicted prices (π)				
Adjusted prediction = $\max\{\text{perc}, \pi\}$				
Job length	2 slots			Surplus = Value - Cost 35 - ? 25 - ? 25 - ?
Deadline values	---	\$35	\$25	
Bundle {1,2}	yes	yes	---	
Bundle {1,3}	yes	---	yes	
Bundle {2,3}	---	yes	yes	

Price Predictors

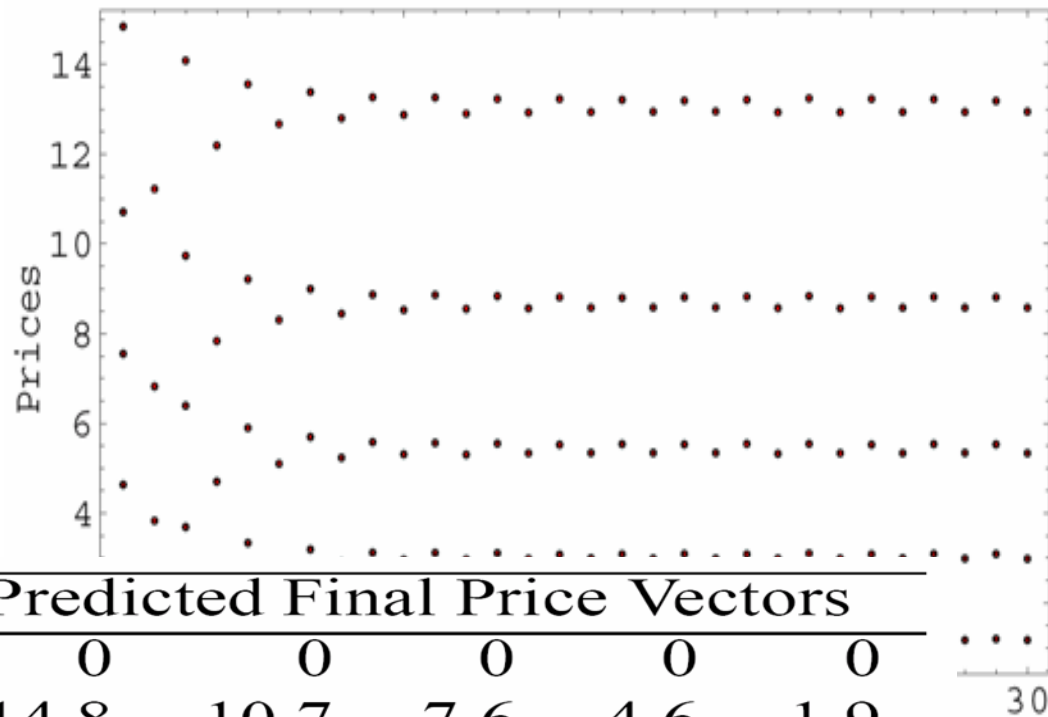
Estimate final prices:

- Perceived prices
- Vector of predicted prices
- Adjusted prediction
- Best bundle
 - maximizes surplus at adjusted predicted prices
 - assumes it will win the whole bundle

Slots	1	2	3	
Current prices (CP)	\$10	\$9	\$5	
Slots the agent is winning	yes	---	---	
Perceived prices	\$10	\$10	\$6	
Vector of predicted prices (π)	\$20	\$15	\$1	
Adjusted prediction = $\max\{\text{perc}, \pi\}$	\$20	\$15	\$6	
Job length	2 slots			Surplus = Value - Cost
Deadline values	---	\$35	\$25	
Bundle {1,2}	yes	yes	---	
Bundle {1,3}	yes	---	yes	
Bundle {2,3}	---	yes	yes	
				35 - 35 = \$0
				25 - 26 = -\$1
				25 - 21 = \$4

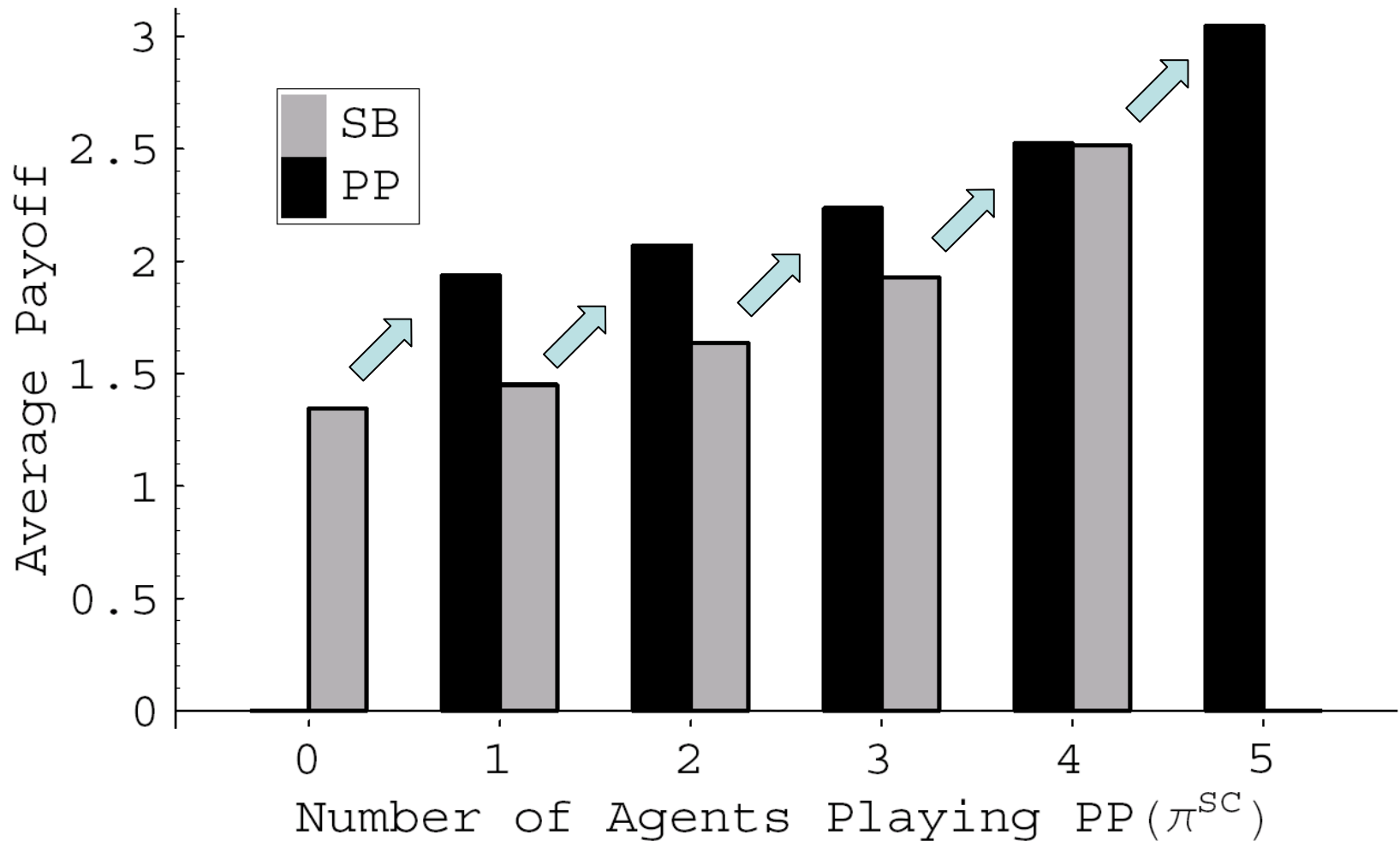
Predictors

- SB = Straightforward Bidding = $PP(0)$
- BL = Baseline = Predict average prices for SB agents
- SC = Self-Confirming
- ECE = Expected Competitive Equilibrium
- EDCE = Competitive Equilibrium for expected demand

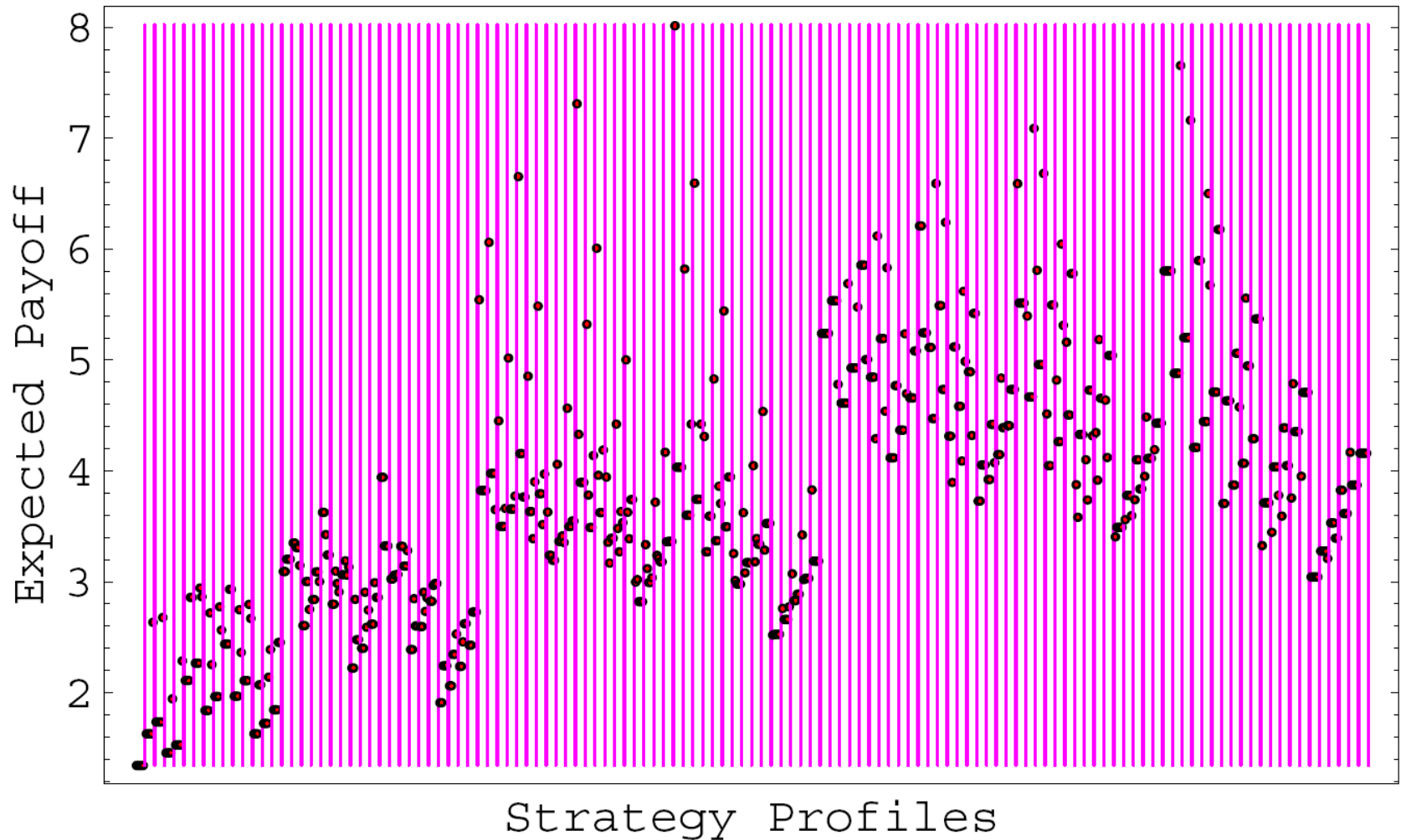


Prediction Methods	Predicted Final Price Vectors				
SB ($=PP(0)$)	0	0	0	0	0
$PP(\pi^{BL})$	14.8	10.7	7.6	4.6	1.9
$PP(\pi^{SC})$	13.0	8.7	5.4	3.0	1.2
$PP(\pi^{ECE})$	26.0	14.2	6.9	2.5	0.3
$PP(\pi^{EDCE})$	20.0	12.0	8.0	2.0	0.0

Example Payoff Matrix (2 strategies)



Payoff Matrix for 5-Strategy Game



Participation-Only Prediction

- Predictors always beat straightforward bidders
- Why does prediction help?
- Decompose behavior of predictor
 - Finding the best bundle
 - Deciding whether to bid on it
- Modified (PO) Predictor:
 - Pick best bundle as per SB
 - Only bid on it if positive surplus at *predicted* prices

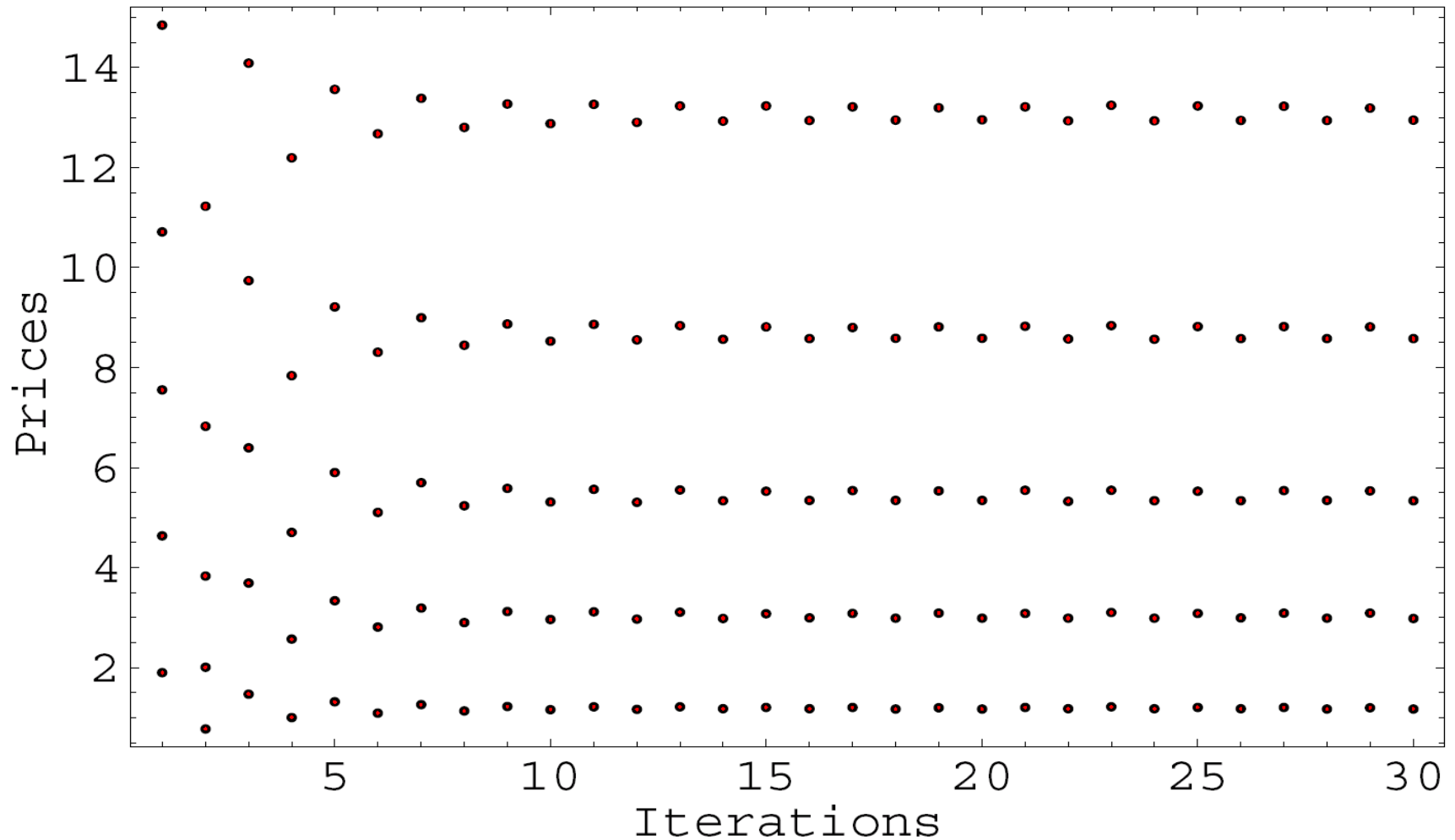
Equilibria and Efficiency Results

- 98% of performance improvement is due to correct choice of participation or not
- SC and EDCE are supported in an equilibrium of 5-predictor game
- Prediction greatly improves agent (buyer) performance with a small efficiency loss (ie, hurts the seller)
 - Buyer surplus three times greater for equilibrium price predictors than all SB agents
 - Market efficiency (aggregate utility as fraction of optimal allocation) drops from 87% to 86%

Conclusion: Price Prediction Helps in SAA

- Price prediction can significantly improve performance by reducing the exposure risk
- Performance depends on the quality of prediction and on how it is used
- Computational game theoretic approach to assessing strategies
- Results specific to particular scheduling domain
- But this is the best known strategic approach to bidding in any Simultaneous Ascending Auctions
- We expect price distribution predictors to perform much better...

Convergence to Self-Confirming Price Predictions



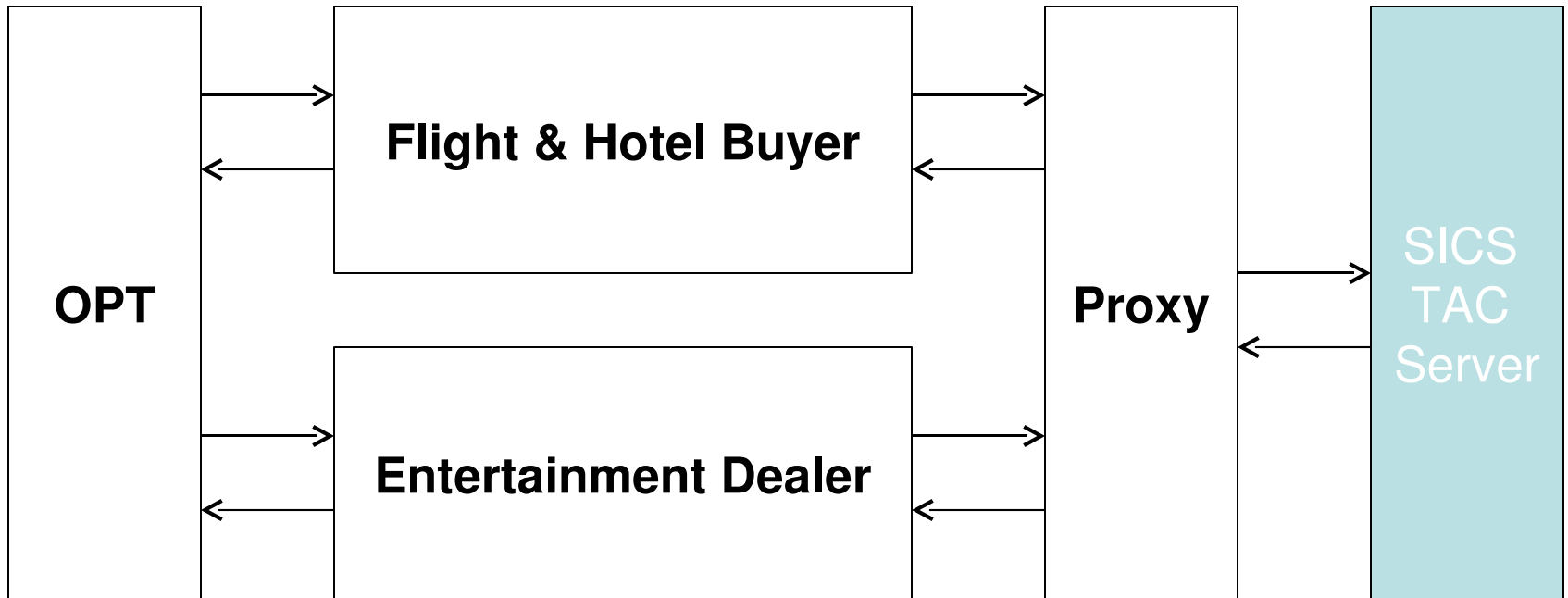
Equilibria and Efficiency Results Table

Games (i.e., strategy sets)	Equilibrium Profiles	% Eff.	Payoff	Average Final Price Vectors				
$\{\text{SB}, \text{PP}(\pi^{BL})\}$	all $\text{PP}(\pi^{BL})$	86	4.15	11.2	6.8	3.8	2.0	0.77
$\{\text{SB}, \text{PP}(\pi^{BL}) \text{ w/ P.O.}\}$	all $\text{PP}(\pi^{BL})$ w/ P.O.	85	4.07	11.8	6.9	3.7	1.7	0.58
$\{\text{SB}, \text{PP}(\pi^{SC})\}$	all $\text{PP}(\pi^{SC})$	88	3.05	13.0	8.7	5.4	3.0	1.17
$\{\text{SB}, \text{PP}(\pi^{BL}), \text{PP}(\pi^{SC}), \text{PP}(\pi^{ECE}), \text{PP}(\pi^{EDCE})\}$	0.45 SC, 0.55 EDCE	86	4.25	10.6	6.5	4.0	2.2	0.91
Additional Profiles								
	all SB	87	1.35	14.8	10.7	7.6	4.6	1.90
	all $\text{PP}(\pi^{ECE})$	74	5.80	4.7	2.1	1.7	1.2	0.55
	all $\text{PP}(\pi^{EDCE})$	83	5.24	8.1	4.5	2.7	1.6	0.70

Walverine

a TAC-02 Agent from the
University of Michigan

Architecture



Flight & Hotel Loop

- Initial
 - Get flight prices
 - Initial predict
 - Client-by-client best-trip optimization
 - Buy flights
- Starting at 3:00, each minute:
 - Get quotes, transactions
 - Price prediction
 - Optimal package, buy flights if nec.
 - Get marginal values
 - Construct hotel bids

Price Prediction



Given initial flight prices, calculate
Walrasian competitive equilibrium
hotel prices.

Premises:

- Trip choices driven in large part by relative flight prices.
- Aggregate behavior reasonably approximated by competitive model.

Calculating Competitive Eq.

- Iterative price adjustment (tatonnement):

$$p_{t+1} \leftarrow p_t + \alpha_t [x(p_t) - 16]$$

where $x(p_t)$ is aggregate demand at hypothetical prices

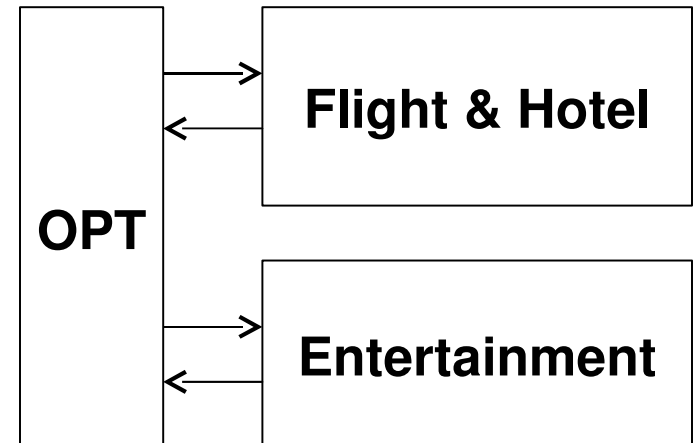
- Demand estimation
 - For our own clients (8), calculate hotel rooms demanded for best package at hypothetical prices
 - For other agents' clients (56), employ **analytic expression for expected demand** based on client preference distribution
- Mid-game:
 - Employ quote/closing as price floors
 - Fix own demand at holdings for closed hotels

Hedging

- Equilibrium method yields point estimate, decisions highly sensitive.
- “Outlier probability”
 - Represents likelihood that prediction is wrong for a given hotel.
 - Outlier prediction defined as $\max(2p, 400)$.
 - Walverine uses 0.06 per open hotel.
- Results in hedged package choices, hotel valuations.

Optimizer

- Integer linear program representing optimal package allocation
- Inputs: prices (actual, predicted), holdings
 - Reported separately by hotel, entertainment modules
- Outputs:
 - Optimal package
 - Marginal values (each unit)
 - Hedged marginal values



Hotel Bidding

- Compute hedged MV given predicted prices
- Compute *optimal shading*
 - Separately for each unit
 - Based on analytic model of other clients' MV distribution (similar to demand calc. approach)
 - Maximize bid value, accounting for:
 - Prob of winning unit & setting price
 - Prob of winning & not setting
 - Expected price if no bid
 - Number of other units affected
- Adjust optimal shades for BTQ rule

} conditional on ASK

Entertainment Dealing

- Derived a trading policy via Q-learning.
- Action is bid
 - to buy/sell unit in given entertainment auction
 - represented as offset from marginal value
- State space:
 - Game time, MV, holdings, Bid/Ask, day
 - Coarse distinctions, still 12852 states
- Rewards: Entertainment cash flow + fun bonus

Learning to Trade

- Two giant Q-tables shared by all auctions (one each for days 1/4, 2/3)
- Played various policies, gathering transition and reward data
 - livingagents, “Exploit”, “Explore”
 - Games on SICS/own servers, variously populated
 - 14839 total games (\times 12 auctions)

Summary: Walverine

- Price prediction based on competitive eq.
- Model-based optimal bidding.
- Q-Learned entertainment trading policy.

Flight & Hotel Buyer

Data Free!

Entertainment Dealer

Model Free!

Online Auction Environments

- Auctions are efficient mechanisms for allocating resources
- Online auction space growing
 - Consumer – ebay, amazon
 - B2B
 - Electronic Trading Network (finance)
- Agents aid in the automation of trade in such auctions

Flights

- Agents are buyers
- One flight per day each way
- Prices determined by stochastic process
 - Random walk with upward drift

→ In expectation, flight prices increase over time, but different auctions increase at different rates

Hotels

- Two hotels (Towers, Shanties)
- Each with fixed number (16) of rooms available per day.
- Sold in simultaneous ascending 16th price auctions.
- To avoid sniping: One hotel closes randomly every minute

Entertainment Tickets

- Fixed pool of tickets for Museum, Amusement Park, Alligator Wrestling, by date, divided among agents.
- Clients have different preferences for event type.
- Agents trade among themselves through Continuous Double Auctions (e.g., stock market)

Agent Objectives

- Maximize total “profit”:
[sum over clients: trip utility] minus expenditures
- Client preferences: arrive/depart days, hotel premium, entertainment prefs
- Feasible trip: round trip airline, hotel room for interval
- Trip utility:
 - zero if infeasible
 - if feasible... (next)

Feasible Trip Utility

- Trip utility =
 $1000 - \text{travel penalty} + \text{hotel bonus} + \text{fun bonus}$
- travel penalty = 100 per day deviation
- hotel bonus = $\{1 \text{ if Towers}\} \times \text{premium}$
- fun bonus =
Sum over types: $\{1 \text{ if ticket}\} \times \text{value}$

HotelAgent behavior

- First 30 seconds:
 - Generate hotel price predictions
 - Calculate optimal package, hedging for price volatility
 - Purchase flights in optimal package
- Before each hotel closing
 - Update price predictions
 - Calculate hedged marginal values of hotels
 - Calculate/submit an optimal bid for each hotel

Price Prediction



- Given
 - Distribution over client preferences
 - Assumptions about other agents' bidding
 - Known initial flight prices
- Compute Walrasian equilibrium prices for hotels
 - Prices for which supply meets expected demand

Hedging

- Equilibrium method yields point estimate, decisions highly sensitive.
- “Outlier probability”
 - Represents likelihood that prediction is wrong for a given hotel.
- Calculate hedged package choices, hotel valuations.

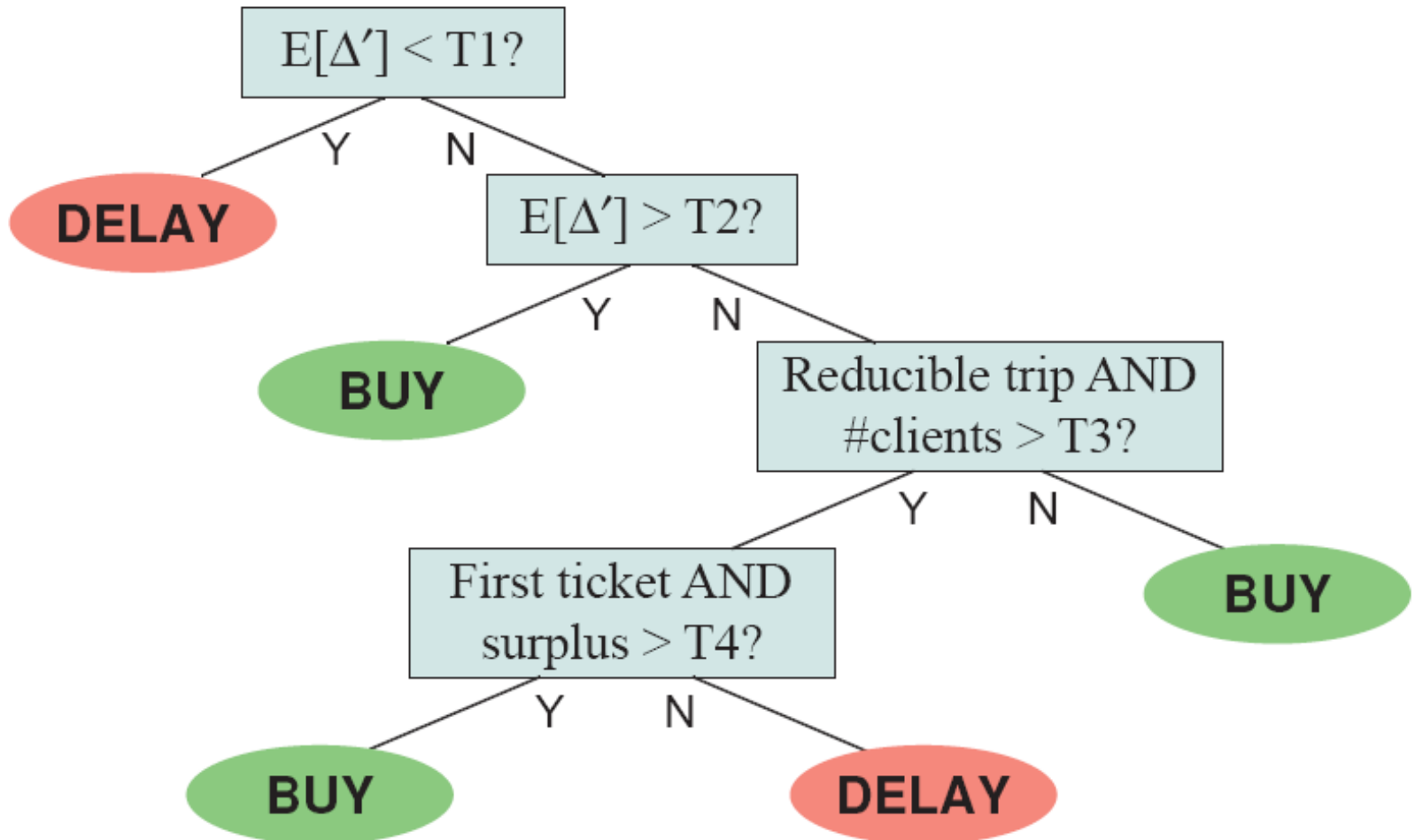
Hotel Bidding

- Compute hedged MV given predicted prices
- Compute *optimal bids*
 - Separately for each unit
 - Based on analytic model of other clients' MV distribution (similar to demand calc. approach)
 - Maximize bid value, accounting for:
 - Probability of winning
 - Expected impact on closing price

Walverine's Entertainment bidding:

- Entertainment accounts for ~40% of Walverine's total score.
- Our approach: Completely model-free.
- Policy derived through Q-learning algorithm.
- Reward: sum of cash-flow plus fun bonus.
- State space defined in terms of six dimensions: time, bid/ask quotes, ticket holdings, and marginal values to buy/sell.
- Actions: bid in terms of offsets from marginal value.

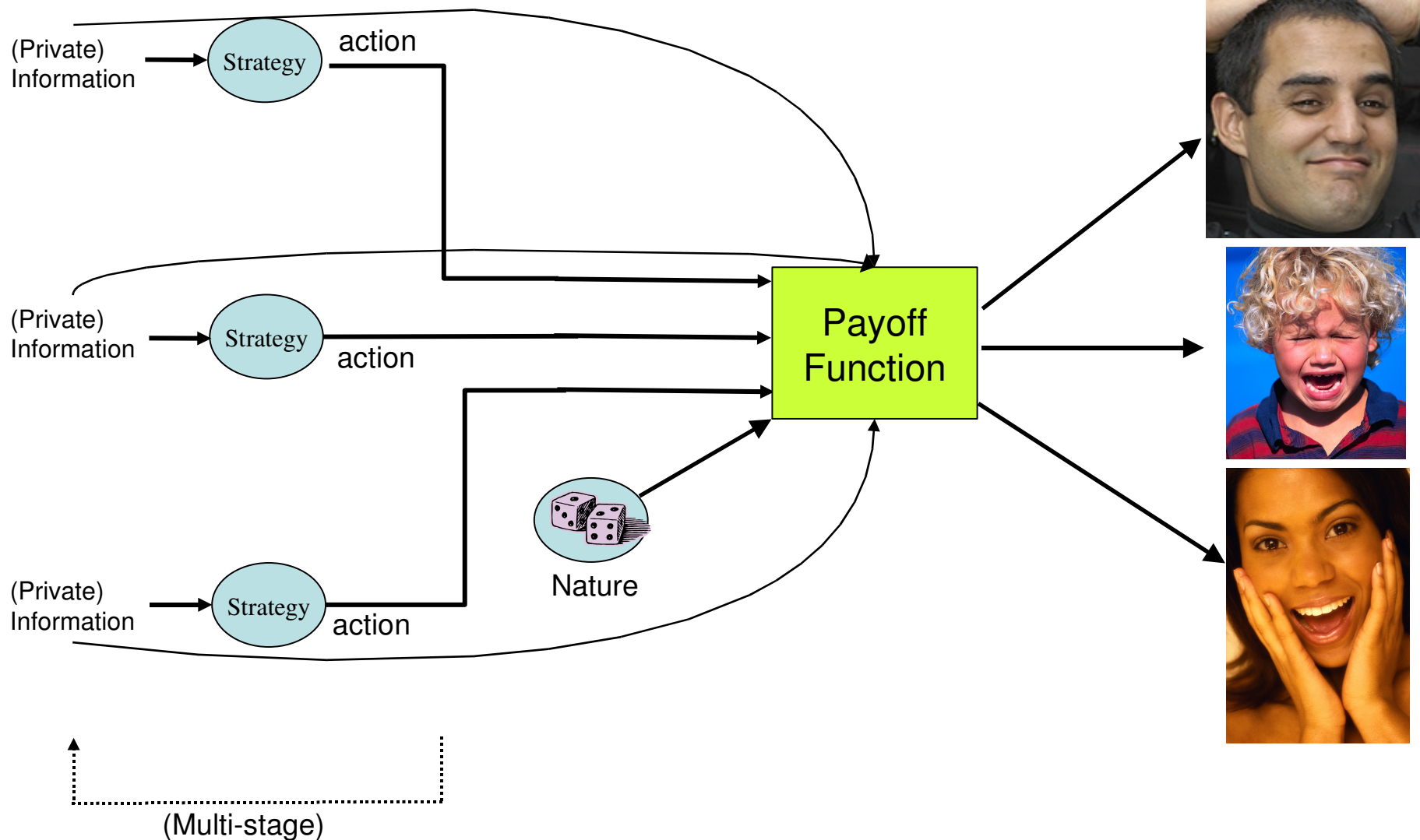
Decision Tree for Flight Buying



How to Bid in Ebay

- Single good with independent private values: Bid your max!
- Multiple goods
 - Need to know the number of bidders and type distribution (a probability distribution over possible valuation functions for bundles of goods)
 - Compute self-confirming distribution prediction

Game Theory Primer



Randomness from Nature



Generating Trading Agent Strategies:

Analytic and Empirical Methods for Infinite and Large Games

Daniel Reeves

2006 Feb 23

Coauthors:

Michael Wellman, Jeffrey MacKie-Mason,
Anna Osepayshvili, Kevin Lochner,
Shih-Fen Cheng, Rahul Suri,
Yevgeniy Vorobeychik, and Maxim Rytin



“So, you’re all presumably here to learn how to generate trading agent strategies. I’m going to spend more time on the analytic methods for infinite games since it’s my baby but I’ll also tell you about the empirical methods for Large Games.

Everything in this presentation is joint work with my advisor, Mike Wellman.

Much of the empirical game methodology is joint work with Mike and Jeff MacKie-Mason and Anya Osepayshvili.

Much of our published work on the Trading Agent Competition (one of the killer apps for our empirical game methodology) is joint work with Mike, Kevin Lochner, Shih-Fen Cheng, Rahul Suri, and Eugene Vorobeychik. Maxim Rytin helped with some of the proofs of some of the theoretical results in the empirical methods section of the talk.”

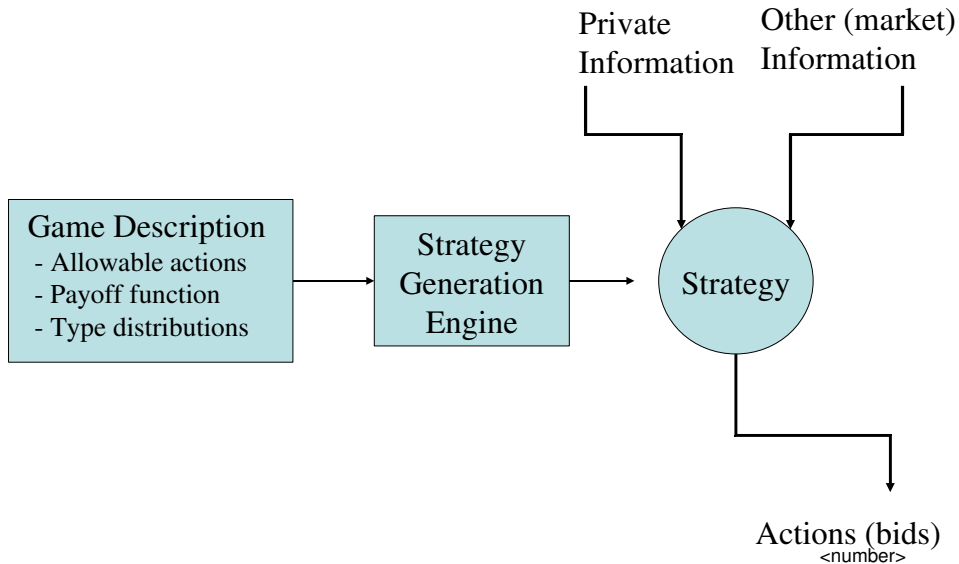
Prefs: doch, players who don’t know what they want don’t last long

Competitiveness assumption: fine, represent advertisers by a demand curve; yahoo and google are the agents

Mike’s presentations I might want to get ideas from:

- TAC Price Prediction
- ppsaa at UAI
- dexter presentation with sunk-awareness results
- hiergame at AAAI

Motivation



“First, the big picture: my grand research vision and the ultimate goal of my thesis work is a Strategy Generation Engine that can read in a description of a game (consisting of game rules and distributions from which private information is drawn) and can output a strategy which is a program that takes actions based on private information and observations about other agent actions. So in a sense the goal is a program that writes programs that play games.

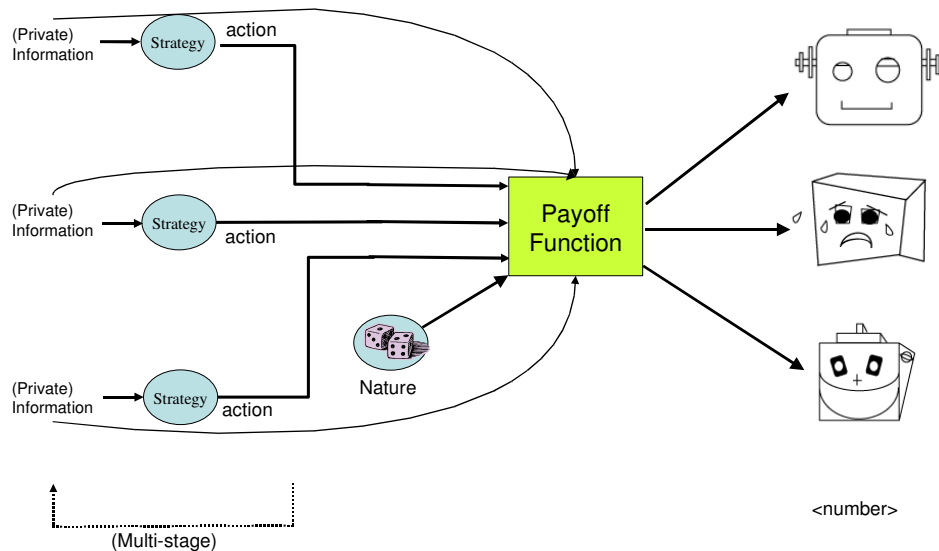
And although I'll focus almost entirely on games that involve auctions of one kind or another, this framework is applicable to more general kinds of games.

But before I say any more about games, let me say what I mean by a game: ...”

YACC analogy:

- * automate what used to be the hard part
- * input high-level spec, turn a crank

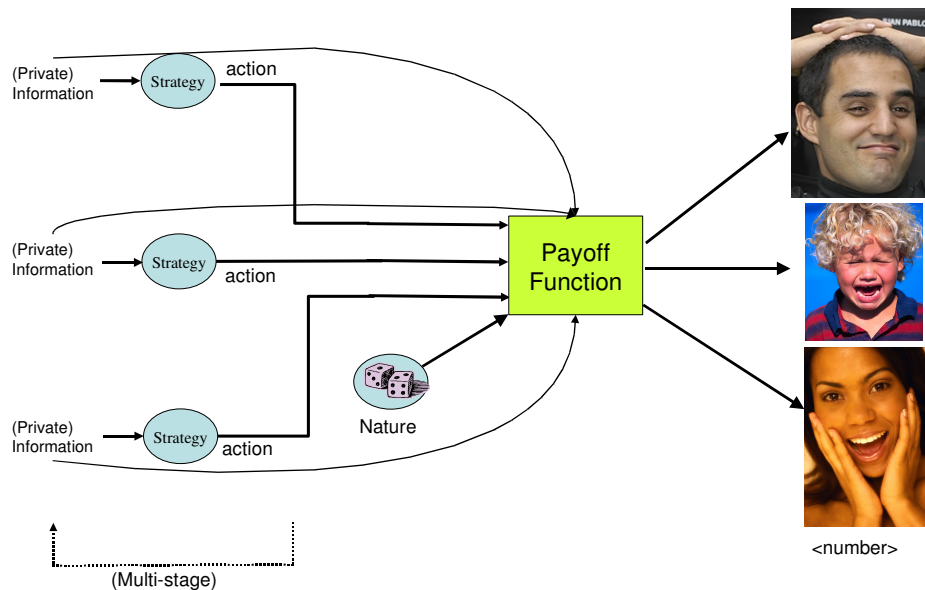
Game Theory Primer



“...it’s really any circumscribed interaction between multiple agents where each agent is trying to maximize an objective function that depends on the interplay of the agents’ actions. The agents here are captured by their strategies—the output of the strategy generation engine. I’ve depicted here a one-shot game. In a multi-stage game, agents perform actions and then learn something about the other agents’ actions (that becoming part of their information) before performing the next action. The whole eventual action history then, plus private information, informs the payoff function. We assume a finite number of rounds. By infinite game I’ll mean that there are infinitely many possible actions—for example, bidding a real number. If there’s any inherent randomness in the game, that’s captured by a dummy player, Nature. The players’ private information is also known as their types or preferences. For example, in poker your hand is your type. In an auction, it would be your valuation of the goods being sold. The output of the payoff function is of course the payoffs, or utilities, of each player. (agent2 didn’t fare so well here) If each agent only has one possible type then this would be a game of complete information, but in general we’re talking about incomplete information games. This captures the case that I may not know what your true payoff function is, but we assume that if only I knew your private information—your type—then I would know. So we always assume that that global payoff function as well as the probability distribution from which the types are drawn are common knowledge. That’s a hairier assumption than it sounds, and if you’re not sure why ask me about the blue-eyed monks problem afterwards. But otherwise, this can be a quite realistic model of real agent interactions. I’m mostly talking about artificial agents but [CLICK] this all applies to humans too. In fact, game theory is even useful in biology, in that case not so much for advising clever squirrels on hoarding strategies but [CLICK] in describing evolved strategic behavior in nature...

[in fact, later in this presentation i’ll describe a method for finding strategies based on a (highly abstracted) biological/evolutionary model.]

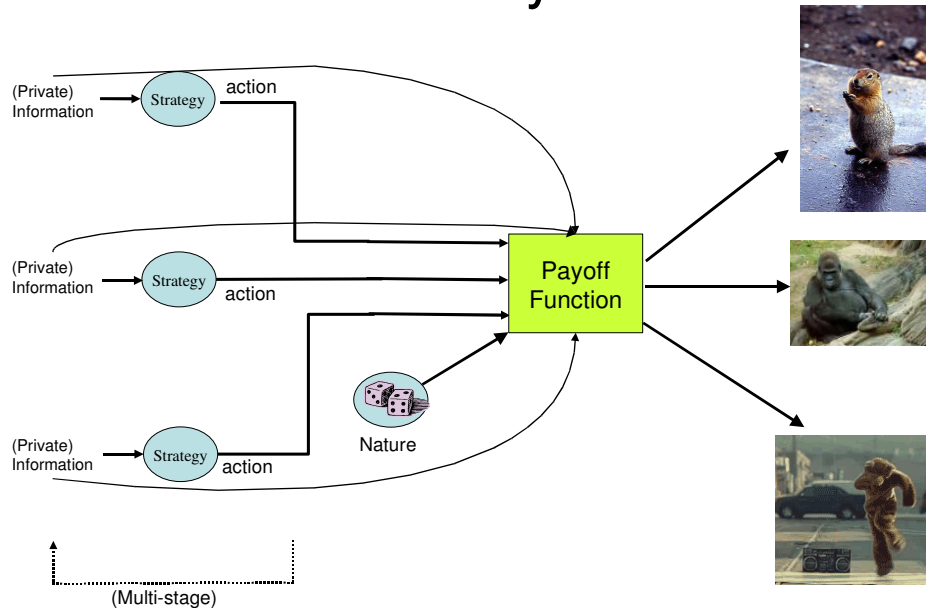
Game Theory Primer



“...it’s really any circumscribed interaction between multiple agents where each agent is trying to maximize an objective function that depends on the interplay of the agents’ actions. The agents here are captured by their strategies—the output of the strategy generation engine. I’ve depicted here a one-shot game. In a multi-stage game, agents perform actions and then learn something about the other agents’ actions (that becoming part of their information) before performing the next action. The whole eventual action history then, plus private information, informs the payoff function. We assume a finite number of rounds. By infinite game I’ll mean that there are infinitely many possible actions—for example, bidding a real number. If there’s any inherent randomness in the game, that’s captured by a dummy player, Nature. The players’ private information is also known as their types or preferences. For example, in poker your hand is your type. In an auction, it would be your valuation of the goods being sold. The output of the payoff function is of course the payoffs, or utilities, of each player. (agent2 didn’t fare so well here) If each agent only has one possible type then this would be a game of complete information, but in general we’re talking about incomplete information games. This captures the case that I may not know what your true payoff function is, but we assume that if only I knew your private information—your type—then I would know. So we always assume that that global payoff function as well as the probability distribution from which the types are drawn are common knowledge. That’s a hairier assumption than it sounds, and if you’re not sure why ask me about the blue-eyed monks problem afterwards. But otherwise, this can be a quite realistic model of real agent interactions. I’m mostly talking about artificial agents but [CLICK] this all applies to humans too. In fact, game theory is even useful in biology, in that case not so much for advising clever squirrels on hoarding strategies but [CLICK] in describing evolved strategic behavior in nature...

[in fact, later in this presentation i’ll describe a method for finding strategies based on a (highly abstracted) biological/evolutionary model.]

Game Theory Primer



“...it’s really any circumscribed interaction between multiple agents where each agent is trying to maximize an objective function that depends on the interplay of the agents’ actions. The agents here are captured by their strategies—the output of the strategy generation engine. I’ve depicted here a one-shot game. In a multi-stage game, agents perform actions and then learn something about the other agents’ actions (that becoming part of their information) before performing the next action. The whole eventual action history then, plus private information, informs the payoff function. We assume a finite number of rounds. By infinite game I’ll mean that there are infinitely many possible actions—for example, bidding a real number. If there’s any inherent randomness in the game, that’s captured by a dummy player, Nature. The players’ private information is also known as their types or preferences. For example, in poker your hand is your type. In an auction, it would be your valuation of the goods being sold. The output of the payoff function is of course the payoffs, or utilities, of each player. (agent2 didn’t fare so well here) If each agent only has one possible type then this would be a game of complete information, but in general we’re talking about incomplete information games. This captures the case that I may not know what your true payoff function is, but we assume that if only I knew your private information—your type—then I would know. So we always assume that that global payoff function as well as the probability distribution from which the types are drawn are common knowledge. That’s a hairier assumption than it sounds, and if you’re not sure why ask me about the blue-eyed monks problem afterwards. But otherwise, this can be a quite realistic model of real agent interactions. I’m mostly talking about artificial agents but [CLICK] this all applies to humans too. In fact, game theory is even useful in biology, in that case not so much for advising clever squirrels on hoarding strategies but [CLICK] in describing evolved strategic behavior in nature...

[in fact, later in this presentation i’ll describe a method for finding strategies based on a (highly abstracted) biological/evolutionary model.]

Game Theory Primer 2

- *Best-Response Strategy* = optimal strategy given known strategies of the other players
- *Nash Equilibrium* = profile of strategies such that each strategy is a best response to the others
- *Bayes-Nash Equilibrium* = generalization of NE to the case of incomplete information, for expected-utility maximizing players
- *Normal-form Game* = defined in terms of strategies
- *Symmetric Game* = no distinct player roles (except Nature)
- *Epsilon of a Profile* = best gain from deviating (0 iff Nash)

<number>

“A few more key definitions. My best-response strategy is my optimal strategy (the one maximizing my payoff) if I knew everyone else’s strategy. A NE is a profile of strategies that are each best responses to the others. Nash himself proved that for any finite game, as long as players can use mixed strategies (that is, randomize among actions) at least one Nash equilibrium must exist. A Bayes-Nash Equilibrium is the natural generalization [TOO TECHNICAL] where players are simultaneously maximizing their *expected* payoffs given the strategies and known type distributions of the other players.

By expressing a possibly multi-stage game by enumerating [describing the space of] the possible strategies, any game can be converted to a one-shot game of complete information, called the normal form of the game.

Finally, almost all the games I’ve studied are symmetric. That means that ex ante (before the types are determined) all the players are identical. For example, we might assume that there’s nothing to distinguish poker players until their hands are dealt. A symmetric profile is simply a homogeneous one—all agents play the same strategy, which in a symmetric game we might well expect they would. A symmetric equilibrium, then, just refers to an equilibrium profile that is symmetric. Nash also proved that symmetric games have symmetric equilibria.

And one last definition: the epsilon of a profile is how much better I could do by playing my best response to the profile instead. So that’s zero iff the profile is an equilibrium and it gives us a measure of how far from equilibrium we are otherwise. (any questions on any of these game theory concepts or terminology at this point?)”

<number>

Outline

- Best-response strategies in one-shot, 2-player, infinite games of incomplete information
- Empirical game methodology for multi-stage, multi-player games (“Taming Monster Games”)
- Taming 2 particular monster games:
 - Simultaneous Ascending Auctions (SAA)
 - Trading Agent Competition (TAC Travel)

<number>

“So, here’s what you’ve really come for: I’ll first describe my analytic approach to generating best-response strategies in a class of one-shot, 2-player infinite games and use that to find Nash equilibria. [WHY IS THIS RELEVANT, IMPORTANT, NOVEL]

That works for many simple games, automating many results published in the game theory and auction theory literature.

For more complex games, I’ll describe our empirical approach and it’s application in particular to the Trading Agent Competition Travel-shopping game.”

Our Class of Infinite Games

- 2-player, one-shot, infinite games of incomplete information
- Piecewise uniform type distributions
- Payoff functions of the form:

$$u(t, a, t', a') = \begin{cases} \theta_1 t + \rho_1 a + \theta'_1 t' + \rho'_1 a' + \phi_1 & \text{if } -\infty < a + \alpha a' < \beta_2 \\ \theta_2 t + \rho_2 a + \theta'_2 t' + \rho'_2 a' + \phi_2 & \text{if } \beta_2 \leq a + \alpha a' \leq \beta_3 \\ \dots & \\ \theta_I t + \rho_I a + \theta'_I t' + \rho'_I a' + \phi_I & \text{if } \beta_I \leq a + \alpha a' \leq +\infty \end{cases}$$

“So here’s our class of 2-player, one-shot, infinite games of incomplete information. The type distribution must be piecewise uniform and payoff functions of my type and my action and the other agent’s type and action are of this form, where all the greek letters are parameters. Note that the piecewise linear restriction is not especially restrictive – you can get as close as you like to an arbitrary function with a piecewise linear approximation.”

Games in our Class

Game	θ	ρ	θ'	ρ'	ϕ	β	α
FPSB	0, 1/2, 1	0, -1/2, -1	0, 0, 0	0, 0, 0	0	0, 0	-1
Vickrey Auction	0, 1/2, 1	0, 0, 0	0, 0, 0	0, -1/2, -1	0	0, 0	-1
Vicious Vickrey Auction	0, $\frac{1-k}{2}$, 1 - k	k, k/2, 0	-k, -k/2, 0	0, $\frac{k-1}{2}$, k - 1	0	0, 0	-1
Supply Chain Game	-1, -1, 0	1, 1, 0	0, 0, 0	0, 0, 0	0	v, v	1
Bargaining Game (seller)	-1, -1, 0	1 - k, 1 - k, 0	0, 0, 0	k, k, 0	0	0, 0	-1
(buyer)	0, 1, 1	0, -k, -k	0, 0, 0	0, 1 - k, 1 - k	0	0, 0	-1
All-Pay Auction	0, 1/2, 1	-1, -1, -1	0, 0, 0	0, 0, 0	0	0, 0	-1
War of Attrition	0, 1/2, 1	-1, -1/2, 0	0, 0, 0	0, -1/2, -1	0, 0, 0	0, 0	-1
* Shared-Good Auction	0, 1/2, 1	0, -1/4, -1/2	0, 0, 0	1/2, 1/4, 0	0	0, 0	-1
* Joint Purchase Auction	0, 1	0, -1/2	0, 0	0, 1/2	0, -C/2	C	1
Subscription Game	0, 1	0, -1	0, 0	0, 0	0, 0	C	1
Contribution Game	0, 1	-1, -1	0, 0	0, 0	0, 0	C	1

$$u(t, a, t', a') =$$

$$\begin{cases} \theta_1 t + \rho_1 a + \theta'_1 t' + \rho'_1 a' + \phi_1 & \text{if } -\infty < a + \alpha a' < \beta_2 \\ \theta_2 t + \rho_2 a + \theta'_2 t' + \rho'_2 a' + \phi_2 & \text{if } \beta_2 \leq a + \alpha a' \leq \beta_3 \\ \dots & \\ \theta_I t + \rho_I a + \theta'_I t' + \rho'_I a' + \phi_I & \text{if } \beta_I \leq a + \alpha a' \leq +\infty \end{cases}$$

“That class of games turns out to be pretty general and captures most 2-player one-shot games you can think of by appropriate settings of those greek letters. Here are the parameter settings for a host of games, some well-known like first-price and 2nd-price sealed-bid auctions, and a couple we made up, namely the shared-good and joint purchase auctions which I’ll describe shortly.”

Payoff for winning in joint purchase auction: $t - a + (a + a' - C)/2$

Payoff we might like to have: $t - a + (a + a' - C) * a / (a + a')$

Piecewise Linear Strategies

$$s(t) = \begin{cases} m_1 t + b_1 & \text{if } -\infty < t \leq c_2 \\ m_2 t + b_2 & \text{if } c_2 < t \leq c_3 \\ \dots & \\ m_{K-1} t + b_{K-1} & \text{if } c_{K-1} < t \leq c_K \\ m_K t + b_K & \text{if } c_K < t \leq +\infty, \end{cases}$$

- Specified by the vectors **c**, **m**, **b**

<number>

“So games can now be described by a set of parameters. If we also specify a piecewise linear strategy, like this, (and remember, in a one-shot game a strategy is just a mapping from type to action) then we’re ready for my main result for this game class...”

Existence and Computation of Piecewise Linear Best Responses

- Theorem 1: Given a payoff function with I regions, an opponent type distribution with cdf F that is piecewise uniform with J pieces, and a piecewise linear strategy function with K pieces, the best response is itself a piecewise linear function with no more than $2(I-1)(J+K-2)$ piece boundaries.

<number>

“That for payoff matrices of THIS [BACKx2] form along with a specified type distribution, and a piecewise linear strategy of THIS [BACK] form, then we can compute in polynomial time the best-response strategy, which will also be a piecewise linear strategy.”

Proof Sketch

- For arbitrary own type t , and opponent type a random variable T , find own action a maximizing $E_T[u(t, a, T, s(T))]$
- (Numerical maximization not applicable due to parameter t)
- Above works out to be a piecewise polynomial in a (parameterized by t)
- For given t , finding optimal a is straightforward
- Remains to find partitioning of type space such that within each type range, optimal action is a linear function of t
- This can be done in polynomial time

<number>

“Just to give you the barest taste of what the proof is like: we have a function that gives our payoff in terms of our type and action and the other agent’s type, which is a random variable from a known distribution. So we need to find our own action, expressed in terms of arbitrary t , our type, that maximizes our expected payoff. In the complete information case, you could then apply any kind of numerical maximization technique to get your best action. But here we end up with a nasty expression involving t . But it turns out the optimal action is always linear in t in any t neighborhood so if we partition the type space in the right way we end up being able to express the optimal action as such-and-such linear function of t when t is in such-and-such range and such-and-such other linear function of t when t ’s in this other range, and so we’ve got a best-response strategy that looks like THIS [BACKx2].

And the great thing about that is it means we can now iterate this best-response algorithm from some arbitrary seed strategy and if it’s lucky enough to converge, ie, reach a fixed point, then we have a strategy that’s a best-response to itself and thus... a Bayes-Nash Equilibrium!”

Cf. Newton iteration, hill climbing

<number>

Example: First-Price Sealed Bid Auction (FPSB)

- Types (valuations) drawn from $U[0,1]$
- Payoff function:

$$u(t, a, a') = \begin{cases} t - a & \text{if } a > a' \\ (t - a)/2 & \text{if } a = a' \\ 0 & \text{otherwise.} \end{cases}$$

- Known Bayes-Nash equilibrium:
 $a(t)=t/2$ (Vickrey, 1961)
- Found in as few as one iteration from a variety of seed strategies

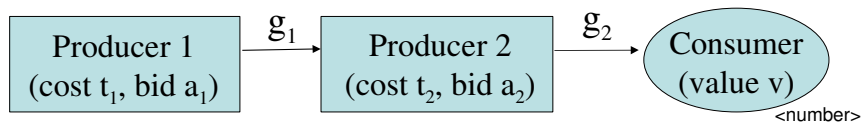
<number>

“A very simple example of a game in our class is the first-price sealed-bid auction. In this game the players’ types are their independent, private valuations for the good, drawn uniformly from $[0,1]$, and the actions are the bids. So your payoff is your valuation minus your bid if your bid is higher, nothing if your bid is lower, and the winner is chosen at random in case of a tie so in expectation your payoff in that case is the average of 0 and $t-a$. So this game has a well-known equilibrium of shading your bid by 1 over the number of players (so $1/2$ here) and my algorithm finds that equilibrium by computing iterated best-responses from most seed strategies – for example, the best-response to truthful bidding is in fact this NE.”

Example: Supply-chain Game

- Producers' Costs $U[0,1]$
- Consumer's Valuation v (known)
- Payoff function:

$$u(t_1, a_1, a_2) = \begin{cases} a_1 - t_1 & \text{if } a_1 + a_2 \leq v \\ 0 & \text{otherwise} \end{cases}$$



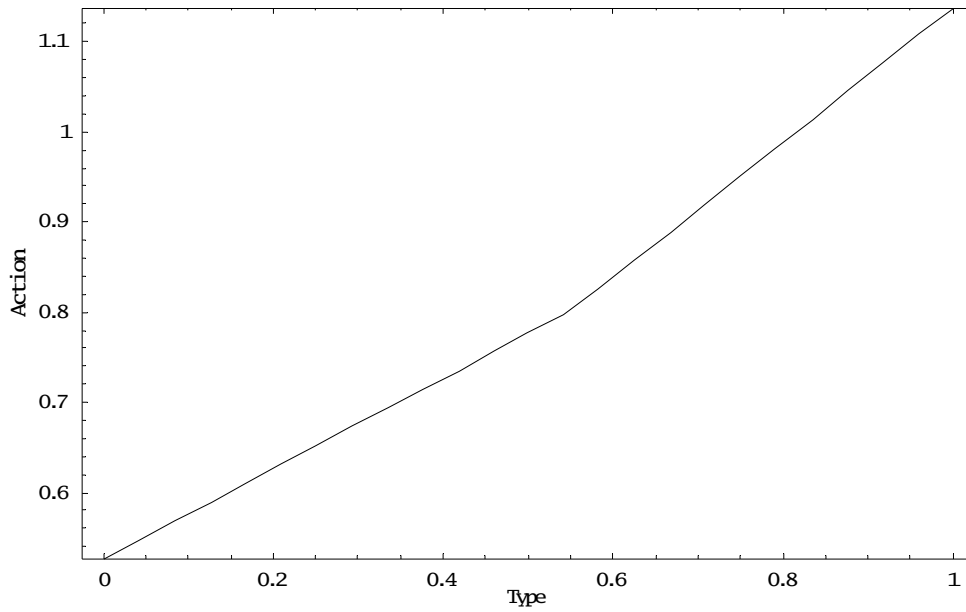
“Here’s another game we call the supply-chain game. [DESCRIBE GAME] This game was studied in a paper on supply chain formation by Bill Walsh and others and they proposed a general strategy for a broader class of supply chain games, which for this game works out to be this...”

2 producers in a supply chain; they bid for being included – the sum of their bids is what the consumer has to pay.

The consumer will pay only if the sum of their bids is less than its valuation v , in which case each producer gets a payoff

of its bid minus its cost. If the sum of bids is higher than v , both agents get nothing.

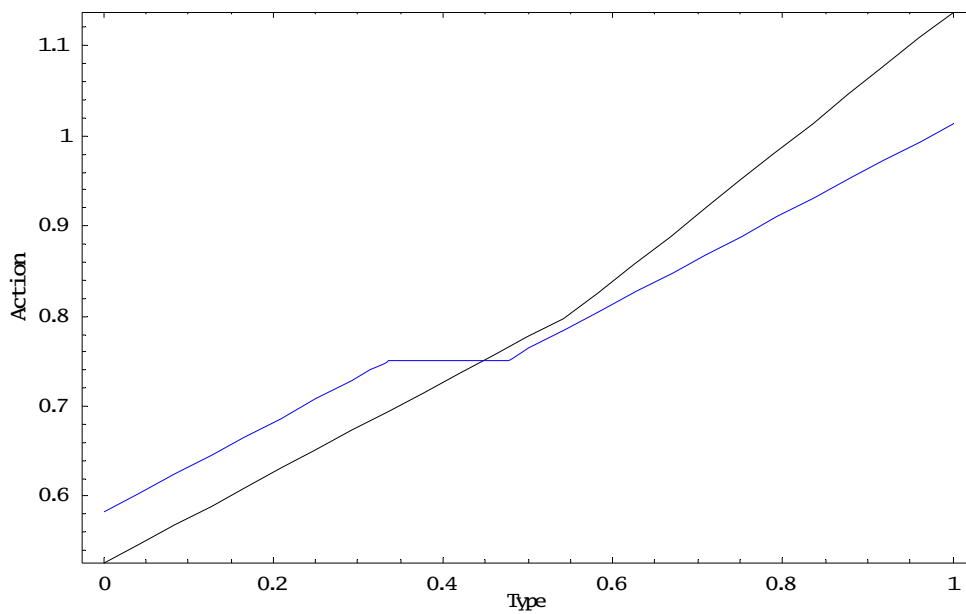
Finding Best Responses



“And here is that strategy, which you can see is of a nice piecewise linear form, so we can feed it to our best-response finder...”

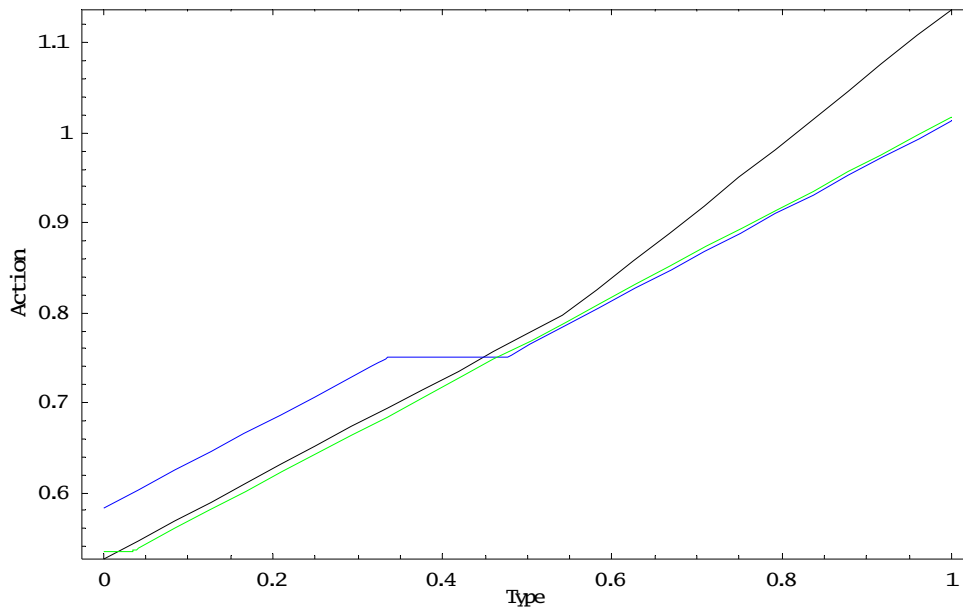
[BOLDER LINES, BIGGER CAPTIONS]

Finding Best Responses



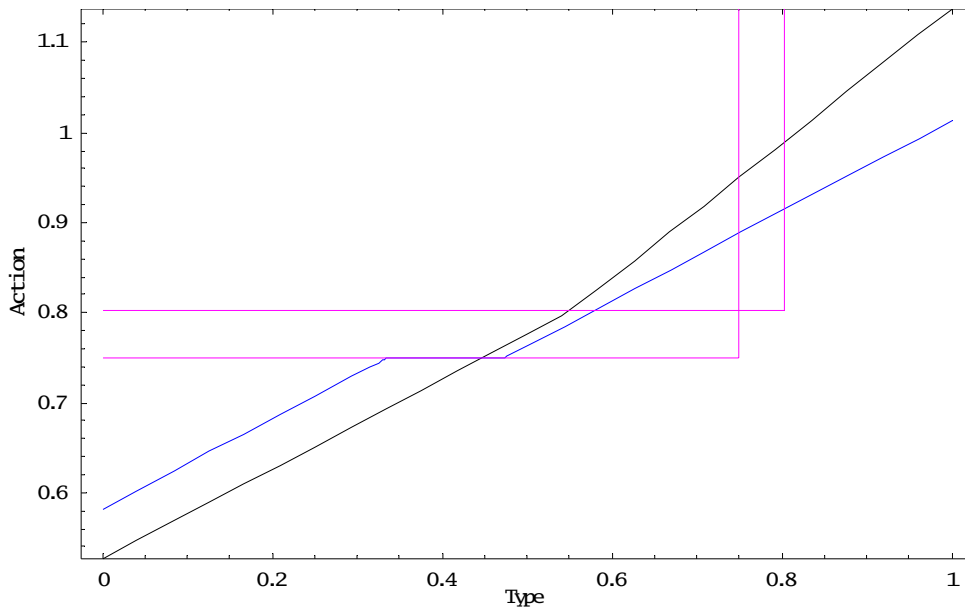
“And we get the strategy in blue.”

Finding Best Responses



“After a few more iterations of best-responses, we seem to be converging to this green strategy which is an equilibrium that we found by hand for this game. And of course, our algorithm confirms that the green strategy is an equilibrium. But that’s not the equilibrium we converge to from the initial seed strategy from Walsh and company...”

Finding Best Responses



“Rather, we end up at this asymmetric equilibrium, which is basically the pair of strategies in which we each ask for half of v (unless that’s not enough given our cost in which case we just kill the deal). It’s pretty easy to see that this is an equilibrium – if I know you’re asking for x then I might as well ask for $v-x$. Any more and we both get nothing. So in fact there are a continuum of equilibria of this form, though iterated best-response always seems to converge to reasonably fair ones, like this, where both agents ask for about half.”

Example: Bargaining Game

- (aka, sealed-bid k-double auction)
- Buyer and seller place bids, transaction happens iff they overlap
- Transaction price is some linear combination of the bids
- Known equilibrium (Chatterjee & Samuelson, 1983) with $k=1/2$ for seller (1) and buyer (2):

$$a_1(t_1) = 2/3t_1 + 1/4$$

$$a_2(t_2) = 2/3t_2 + 1/12$$

- Found in several iterations from truthful bidding

<number>

Another game that's similar to the supply chain game is the bargaining game or sealed-bid k-double auction. This is a very straightforward mechanism in which I want to sell you something so we both write down an offer and if you're willing to buy it for at least what I'm willing to sell it for then I give you the good and you pay me something between our bids. This result was published 20some years ago and my solver finds it automatically when seeded with truthful bidding. And btw, this the only asymmetric game I'll talk about today but asymmetry turns out to be easy to deal with at least in the 2-player case.

Myerson and Satterthwaite: pick 2: BB, IR, Eff in BNE

Example: Joint Purchase Auction

- Variants: contribution/subscription games
- 2 agents want to jointly acquire a good costing C
- Mechanism: simultaneously offer contributions; buy iff $\text{sum} > C$ and split the excess ($\text{sum} - C$) evenly
- Nash: $\frac{2}{3}t + \frac{C}{4} - \frac{1}{6}$

<number>

“Here’s another game. This comes under the heading of public good or provision point mechanisms. I’ll describe a variation that seems to not have been solved in the literature before: namely 2 agents want to jointly acquire a good that costs C for which they have private valuations; they submit a sealed offer and the good gets bought if and only if the sum of their offers exceeds the cost, in which case the excess is split and given back. There are variations where the agents don’t get the excess back, or where they don’t get their contributions back in the case that not enough money is collected. Both of those variations are in the literature and they have very different equilibria from this one, which bears some similarity to the known equilibrium for the bilateral bargaining game [sealed double auction], which my algorithm also finds.

Also, I’ve suggested that most 2-player one-shot games are in the class of games that this algorithm works for but here’s an example of a potentially useful one that’s not, namely the variation on the joint purchase auction where we split the excess in proportion to how much we contributed [BACK TO SLIDE 7 (this is 17)].”

Example: Shared-Good Auction

- New mechanism, similar to the divorce-settlement game; undoes joint-purchase
- Agents place bids for a good they currently share, valuations $\sim U[A, B]$
- High bidder gets the good and pays half its bid to the low bidder in compensation

$$u(t, a, a') = \begin{cases} t - a/2 & \text{if } a > a' \\ a'/2 & \text{otherwise} \end{cases} \quad \text{<number>}$$

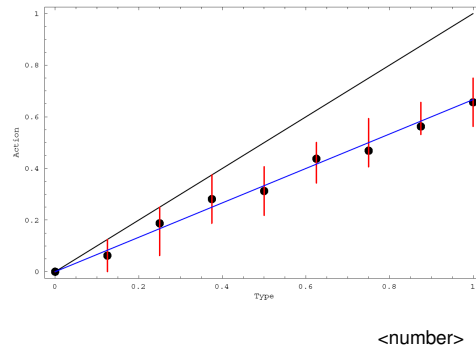
“Now suppose we’ve jointly acquired a good and but now one of us is leaving town and we can’t split the good in half so we need one of us to buy the other out. My friend and colleague Kevin Lochner and I came up with this mechanism originally to decide which of two roommates should get the big bedroom and for how much more rent. Since then we’ve also used it to allocate undesirable tasks for which we had joint responsibility [SO]. The auction rules specify that each agent submits a sealed bid as usual and the high bidder gets the good but pays half their bid to the loser in compensation.”

Parking tickets, skis on ebay (so this is can be used for bid collusion, which is a whole other can of worms).

Equilibrium in Shared-Good Auction

$$a(t) = \frac{2t + A}{3}$$

- Found in one iteration from truthful bidding (for any specific [A,B])



“Again, iterated best-response quickly converges to an equilibrium which is to shade your bid down a third of the way to the lowest possible valuation.

Of course a nice thing about having this game solver for humans using these mechanisms is that we can leverage the revelation principle and then play the modified game where they submit their types to the solver which plays the NE on their behalf. So two people playing the shared-good auction, assuming they believe the type distribution, can simply submit their true types without strategizing. This also mitigates the problem that we have no guarantee the equilibria we find are unique [focal].

By the way, the plotted points with error bars here are just from verifying the analytic algorithm with Monte Carlo simulations.”

Example: Vicious Vickrey Auction

- Generalization of a Vickrey Auction (Brandt & Weiss, 2001) to allow for disutility from opponent's utility (eg, business competitors)

$$u(t, a, t', a') = \begin{cases} (1 - k)(t - a') & \text{if } a > a' \\ ((1 - k)(t - a') - k(t' - a))/2 & \text{if } a = a' \\ -k(t' - a) & \text{otherwise.} \end{cases}$$

- Brandt & Weiss consider only the complete information version

<number>

One other game I wanted to mention is a variant of the Vickrey auction. The interesting thing about a Vickrey Auction – or 2nd price auction because the winner pays the 2nd highest – is that truthful bidding is a dominant strategy (dom = BR to everything). So it's reassuring that my game solver returns truthful bidding for every seed strategy. But the variant is called vicious vickrey because it adds a term to the payoff function for my utility for your disutility. Like if we're competitors, I don't want to only maximize my profit, I want to minimize yours. So now I want to bid more than my valuation in hopes that if I lose I'll at least cause you to pay more.

Equilibrium in Vicious Vickrey

- $a(t) = (k+t)/(k+1)$
- Reduces to truthful bidding for the standard Vickrey Auction ($k=0$)
- Iterated best-response solver finds this equilibrium (for specific values of k) within several iterations from a variety of seed strategies

<number>

And here's the equilibrium strategy.

Conclusions:

Best-Response Solver

- First algorithm for finding best-response strategies in a broad class of infinite games of incomplete information
- Confirms known equilibria (eg, FPSB), confirms equilibria we derive here (Supply-Chain game), discovers equilibria in new games (eg, Joint Purchase and Shared-good auction)
- Goal: characterize the class of games for which iterated best-response converges

<number>

“To conclude the first part of this presentation, we have an algorithm for computing best-response strategies in a broad class of 2-player infinite games of incomplete information.

We’ve used this to confirm many known equilibria, confirm some we’ve derived by hand, and discovered equilibria in new games, namely the joint-purchase and shared-good auctions.

The most interesting and useful aspect of this work is not the actual best-response finding (though that has its uses, like automating proofs of equilibria) but the iterating of that to find NE.

It remains a goal to characterize the class of games, if not our entire class, for which that process converges.”

[GIVE OUTLINE SLIDE WITH PROGRESS BAR]

Taming Monster Games: Overview

- Determining candidate strategies
- Game simulators and brute-force estimation
- Variance reduction for Monte Carlo Sampling
 - Control Variates
 - (Quasi-Random Sampling, Importance Sampling)
- Player Reduction
 - Analyzing Empirical Games
 - Gambit, Amoeba, Replicator Dynamics
 - Sensitivity Analysis
 - PM distributions
 - Confidence bounds on equilibria
- Killer App: Trading Agent Competition (TAC)

<number>

“So what about games that have more than two players or have multiple rounds? In other words, most realistic games? I call them Monster Games because they’re pretty much untouchable by standard game theory. For example, we’re not going to find the NE of poker in the foreseeable future (though Nash himself and others have solved highly abstracted versions of it) or the problem of bidding in multiple simultaneous auctions, like for a set of Yahoo keywords.

I’ll now show you my general empirical game methodology – kind of a collection of computational techniques -- for finding good strategies in games like that.

For all these I’ll use a simple first-price sealed-bid auction as an example and to help verify the methodology.

Broadly, there are several parts to this methodology, but I’ll focus on the first few. The first is about determining a set of candidate strategies – that is, restricting the strategy space. This is the most drastic way in which we cut monster games down to size. Next is game simulation – estimating the restricted game.

Going along with game simulation, I’ll talk about standard variance reduction techniques for Monte Carlo sampling, in particular the method of Control Variates for which we’ve done some controlled experiments with first price sealed bid auctions and also applied in the Trading Agent Competition.

Player Reduction is a kind of complement to reducing the number strategies. It’s another way to radically reduce the size of a game without generally sacrificing too much in terms of solution quality. Once we have an empirical estimate of a game, we can apply various off-the-shelf techniques for solving it.

Finally, I’ll briefly talk about a couple methods for assessing how well the solutions to the empirical game approximate solutions to the underlying game of interest.

I’ll focus on toy examples and tell you briefly about the real monster games that motivated all this.

Reprise: First-Price Sealed-Bid Auction (FPSB)

- Types (valuations) drawn from $U[0,1]$
- Payoff function (2-player case):

$$u(t, a, a') = \begin{cases} t - a & \text{if } a > a' \\ (t - a)/2 & \text{if } a = a' \\ 0 & \text{otherwise.} \end{cases}$$

- Strategy space is set of functions from type to action
- Known Bayes-Nash equilibrium: $a(t) = (n-1)/n t$

<number>

So let's start with our old friend FPSB, which is by no means a monster game; in fact it's been solved. But it has infinite type and strategy spaces and a nonlinear payoff function and so has many of the attributes of games that are analytically intractable.

DESCRIBE (generalization to n players)

Again, the strategy space is the set of functions from type to action, and the unique symmetric equilibrium is to bid $(n-1)/n$ times your type, n being the number of players.

FPSB_n

- Start with a baseline strategy
 - Truthful bidding
- Generalize via parameters
 - Shade factor
 - (Translational parameter, $kt+b$, etc)
- Restricted game:
 - For all agents i in $\{1, \dots, n\}$, bid $k_i t$ for k_i in $[0, 1]$
- Further restrict the game by discretizing k

<number>

So for our purposes now, we'll pretend that we can't deal with such a rich strategy space as that and restrict the strategy space by imposing the constraint that strategies must be of a particular parameterized form. In general, the way we do this is to pick a baseline strategy and introduce parameters that generalize it. For FPSB, the most straightforward strategy (although also the worst possible strategy without actively throwing away money) is truthful bidding – just bidding your valuation. But if we generalize that with a shade factor, k , the strategy space now includes the unique symmetric bayes-nash equilibrium (namely, shading by an n th). If we didn't know that, we could've also included, say, a translational parameter b , or allowed piecewise linear strategies with a fixed number of pieces.

But the point is, even if the baseline strategy is awful, as long as the space of strategies allowed by the parameterization includes smarter strategies (and they need not be identifiable as such a priori) then our methodology has hope of finding them. As we'll see, introducing a shading parameter in FPSB (without knowing a good setting for it) allows us to approximate the unique symmetric equilibrium of underlying infinite game.

First, I'll quickly show some theoretical results for this restricted version of FPSB before showing how we manage when the games are too complicated to admit any such analytic results.

Theoretical Results for FPSBn

- Expected Payoff for playing k_i against everyone else playing k :

$$u_i(k_i, k) = \begin{cases} \frac{1}{2n} & \text{if } k_i = k = 0 \\ \frac{1-k_i}{n+1} \left(\frac{k_i}{k}\right)^{n-1} & \text{if } k_i \leq k \\ \frac{(1-k_i)((n-1)k_i^2 - (n-1)k^2)}{2(n+1)k_i^2} & \text{otherwise.} \end{cases}$$

- Best response to everyone else playing k :

$$BR(k) \equiv \arg \max_{k_i} u_i(k_i, k) = \begin{cases} \text{undefined} & \text{if } k = 0 \\ \xi & \text{if } k < \frac{n-1}{n} \\ \frac{n-1}{n} & \text{if } k \geq \frac{n-1}{n} \end{cases} \quad \text{<number>}$$

First, we derived a closed-form expression for the expected payoff for arbitrary symmetric profiles and unilateral deviations.

Maximizing that wrt to k_i gives a closed-form expression for the best-response, where that ξ is an expression of n and k (anyone want to guess what it looks like? Hint: it starts with a cube root of 3... CLICK).

Anyway, from this [CLICK] we can see that the equilibrium in the unrestricted game is also an equilibrium here, and with a bit more effort we show that it's in fact the unique symmetric equilibrium.

Theoretical Results for FPSBn

- Expected Payoff for playing k_i against everyone else playing k :

$$u_i(k_i, k) = \begin{cases} \frac{1}{2n} & \text{if } k_i = k = 0 \\ \frac{1-k_i}{n+1} \left(\frac{k_i}{k}\right)^{n-1} & \text{if } k_i \leq k \\ \frac{(1-k_i)((n-1)k_i^2 - (n-1)k^2)}{2(n+1)k_i^2} & \text{otherwise.} \end{cases}$$

- Best response to everyone else playing k :

$$\frac{\sqrt[3]{3} \left(k^2 (n^2 - 1) \left(9n + \sqrt{3(n+1)((n-1)k^2 + 27(n+1)) + 9} \right) \right)^{2/3} - 3^{2/3} k^2 (n^2 - 1)}{3(n+1) \sqrt[3]{k^2 (n-1) \left(9n^2 + 18n + (n+1)^{3/2} \sqrt{3(n-1)k^2 + 81(n+1) + 9} \right)}}$$

First, we derived a closed-form expression for the expected payoff for arbitrary symmetric profiles and unilateral deviations.

Maximizing that wrt to k_i gives a closed-form expression for the best-response, where that k_i is an expression of n and k (anyone want to guess what it looks like? Hint: it starts with a cube root of 3... CLICK).

Anyway, from this [CLICK] we can see that the equilibrium in the unrestricted game is also an equilibrium here, and with a bit more effort we show that it's in fact the unique symmetric equilibrium.

Theoretical Results for FPSBn

- Expected Payoff for playing k_i against everyone else playing k :

$$u_i(k_i, k) = \begin{cases} \frac{1}{2n} & \text{if } k_i = k = 0 \\ \frac{1-k_i}{n+1} \left(\frac{k_i}{k}\right)^{n-1} & \text{if } k_i \leq k \\ \frac{(1-k_i)((n-1)k_i^2 - (n-1)k^2)}{2(n+1)k_i^2} & \text{otherwise.} \end{cases}$$

- Best response to everyone else playing k :

$$BR(k) \equiv \arg \max_{k_i} u_i(k_i, k) = \begin{cases} \text{undefined} & \text{if } k = 0 \\ \xi & \text{if } k < \frac{n-1}{n} \\ \frac{n-1}{n} & \text{if } k \geq \frac{n-1}{n} \end{cases} \quad \text{<number>}$$

First, we derived a closed-form expression for the expected payoff for arbitrary symmetric profiles and unilateral deviations.

Maximizing that wrt to k_i gives a closed-form expression for the best-response, where that ξ is an expression of n and k (anyone want to guess what it looks like? Hint: it starts with a cube root of 3... CLICK).

Anyway, from this [CLICK] we can see that the equilibrium in the unrestricted game is also an equilibrium here, and with a bit more effort we show that it's in fact the unique symmetric equilibrium.

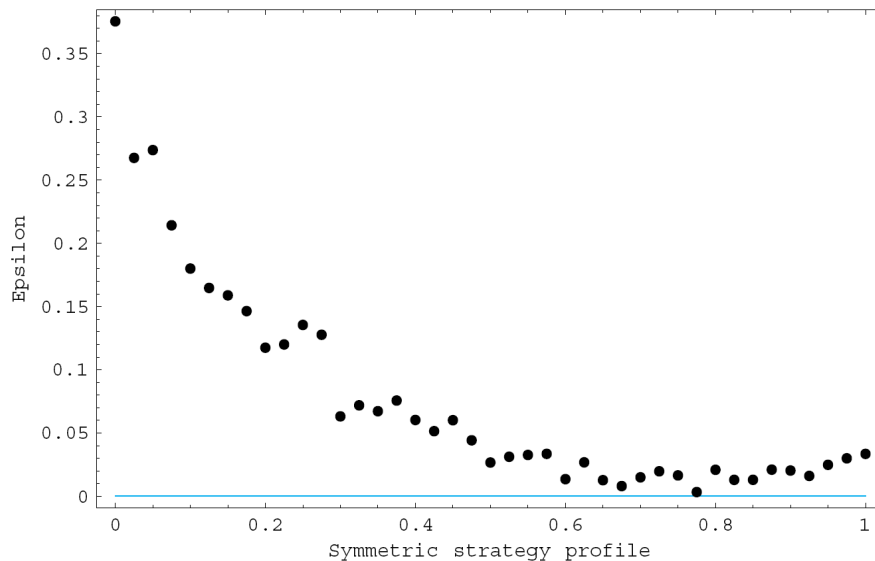
Epsilon Metric for FPSBn

- The epsilon for a profile s is greatest possible gain from deviating (payoff of best unilateral deviation from s minus payoff in s)
- For a symmetric profile (k) in FPSBn:

$$\epsilon_{FPSBn}(k) = \max_x (u_i(x, k)) - u_i(k, k) = \begin{cases} \frac{1}{2} - \frac{1}{2n} & \text{if } k = 0 \\ \frac{(k - \xi)(\xi^2 - k\xi + \xi + k + (\xi - 1)(\xi + k)n)}{2\xi^2(n + 1)} & \text{if } k < \frac{n - 1}{n} \\ \frac{1 - n + k \left(\left(\frac{n - 1}{kn} \right)^n + n - 1 \right)}{n^2 - 1} & \text{otherwise.} \end{cases}$$

We can also derive an expression for the epsilon of arbitrary symmetric profiles in FPSBn, and remember, epsilon means the greatest gain from deviating from a profile.

Game Simulation: FPSB4



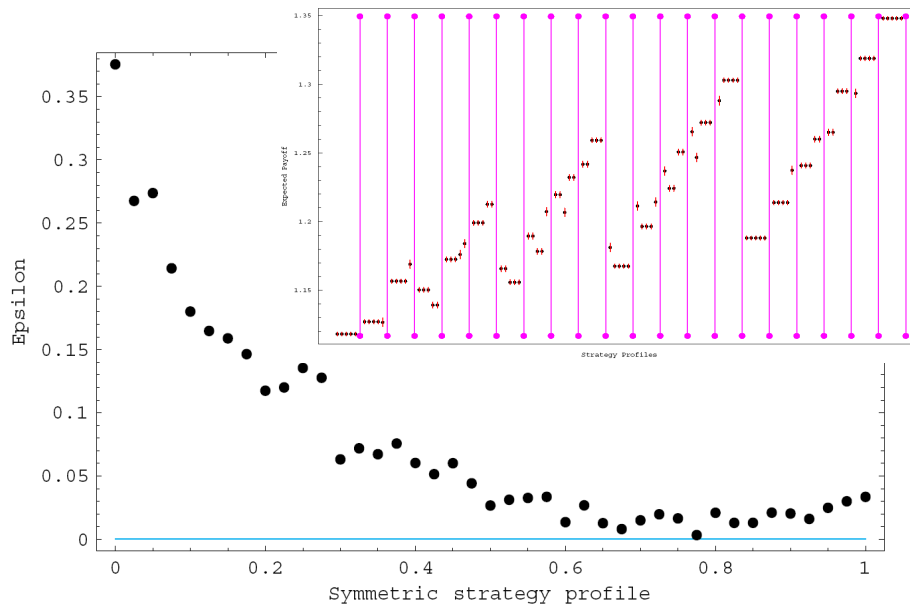
So those results will be nice for comparison but we can't get that sort of thing for the real monster games. Instead, the next step is to further restrict the strategy space by discretizing the parameters. Here, I discretize the strategies in the 4-player restricted FPSB game by 40ths, yielding 135,000 some profiles. I simulated 100 games for each profile to construct the empirical payoff matrix. [CLICK] (We typically represent empirical payoff matrices like this...) This graph shows the epsilons for each symmetric profile, based on that empirical payoff matrix. This is enough to see that there's likely an equilibrium somewhere around .7 something. If we overlay the exact results from the previous slide [CLICK], we can see that this is in fact giving us a reasonable idea of the solution to the underlying game.

(And if you're wondering why all the plotted points are above the exact epsilon it's because all the payoff estimates that the empirical epsilons are based on have sampling noise and since the epsilon calculation takes the max of a bunch of noisy estimates, it will typically find one that's anomalously high. I'll talk about reducing sampling noise next.)

Anyway, to get a more accurate estimate of the solution to the game using brute-force Monte Carlo, I upped the number of samples per profile to 36,000. At this point, we still can't estimate the unique symmetric nash with very high fidelity but it's clear that anything in this range will have very low epsilon in the underlying game.

[100 games per profile and $41+4-1$ choose 4 = 135751 profiles]

Game Simulation: FPSB4



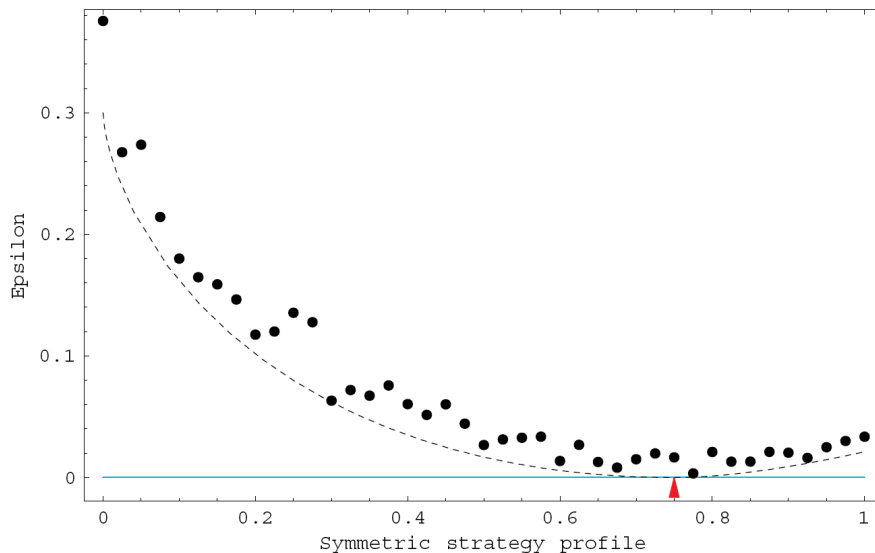
So those results will be nice for comparison but we can't get that sort of thing for the real monster games. Instead, the next step is to further restrict the strategy space by discretizing the parameters. Here, I discretize the strategies in the 4-player restricted FPSB game by 40ths, yielding 135,000 some profiles. I simulated 100 games for each profile to construct the empirical payoff matrix. [CLICK] (We typically represent empirical payoff matrices like this...) This graph shows the epsilons for each symmetric profile, based on that empirical payoff matrix. This is enough to see that there's likely an equilibrium somewhere around .7 something. If we overlay the exact results from the previous slide [CLICK], we can see that this is in fact giving us a reasonable idea of the solution to the underlying game.

(And if you're wondering why all the plotted points are above the exact epsilon it's because all the payoff estimates that the empirical epsilons are based on have sampling noise and since the epsilon calculation takes the max of a bunch of noisy estimates, it will typically find one that's anomalously high. I'll talk about reducing sampling noise next.)

Anyway, to get a more accurate estimate of the solution to the game using brute-force Monte Carlo, I upped the number of samples per profile to 36,000. At this point, we still can't estimate the unique symmetric nash with very high fidelity but it's clear that anything in this range will have very low epsilon in the underlying game.

[100 games per profile and $41+4-1$ choose 4 = 135751 profiles]

Game Simulation: FPSB4



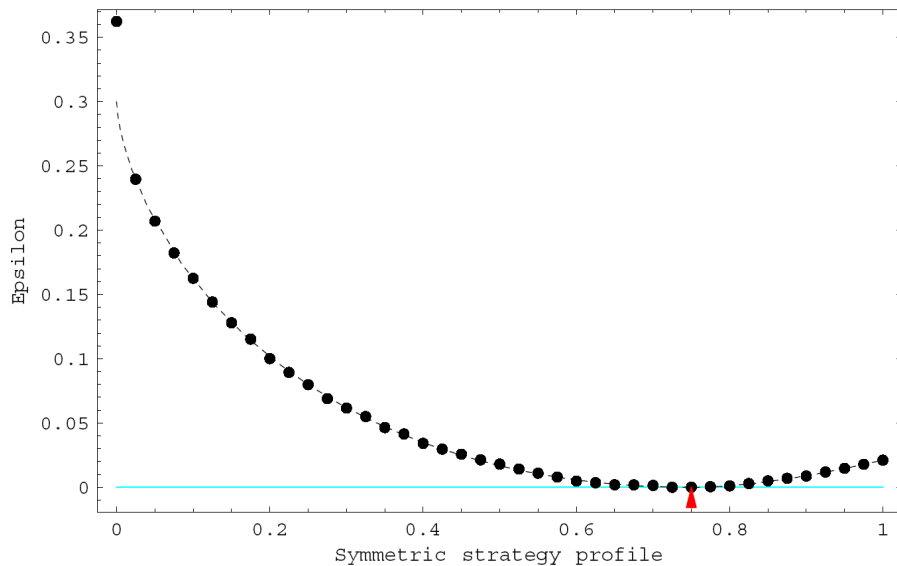
So those results will be nice for comparison but we can't get that sort of thing for the real monster games. Instead, the next step is to further restrict the strategy space by discretizing the parameters. Here, I discretize the strategies in the 4-player restricted FPSB game by 40ths, yielding 135,000 some profiles. I simulated 100 games for each profile to construct the empirical payoff matrix. [CLICK] (We typically represent empirical payoff matrices like this...) This graph shows the epsilons for each symmetric profile, based on that empirical payoff matrix. This is enough to see that there's likely an equilibrium somewhere around .7 something. If we overlay the exact results from the previous slide [CLICK], we can see that this is in fact giving us a reasonable idea of the solution to the underlying game.

(And if you're wondering why all the plotted points are above the exact epsilon it's because all the payoff estimates that the empirical epsilons are based on have sampling noise and since the epsilon calculation takes the max of a bunch of noisy estimates, it will typically find one that's anomalously high. I'll talk about reducing sampling noise next.)

Anyway, to get a more accurate estimate of the solution to the game using brute-force Monte Carlo, I upped the number of samples per profile to 36,000. At this point, we still can't estimate the unique symmetric nash with very high fidelity but it's clear that anything in this range will have very low epsilon in the underlying game.

[100 games per profile and $41+4-1$ choose 4 = 135751 profiles]

Game Simulation: FPSB4



So those results will be nice for comparison but we can't get that sort of thing for the real monster games. Instead, the next step is to further restrict the strategy space by discretizing the parameters. Here, I discretize the strategies in the 4-player restricted FPSB game by 40ths, yielding 135,000 some profiles. I simulated 100 games for each profile to construct the empirical payoff matrix. [CLICK] (We typically represent empirical payoff matrices like this...) This graph shows the epsilons for each symmetric profile, based on that empirical payoff matrix. This is enough to see that there's likely an equilibrium somewhere around .7 something. If we overlay the exact results from the previous slide [CLICK], we can see that this is in fact giving us a reasonable idea of the solution to the underlying game.

(And if you're wondering why all the plotted points are above the exact epsilon it's because all the payoff estimates that the empirical epsilons are based on have sampling noise and since the epsilon calculation takes the max of a bunch of noisy estimates, it will typically find one that's anomalously high. I'll talk about reducing sampling noise next.)

Anyway, to get a more accurate estimate of the solution to the game using brute-force Monte Carlo, I upped the number of samples per profile to 36,000. At this point, we still can't estimate the unique symmetric nash with very high fidelity but it's clear that anything in this range will have very low epsilon in the underlying game.

[100 games per profile and $41+4-1$ choose 4 = 135751 profiles]

Control Variates

- Variance reduction by *adjusting for luck*
- FPSB: higher valuation means higher expected payoff
- Suppose $g(t)$ estimates payoffs; adjust sampled payoffs by subtracting $g(t) - E[g(t)]$

$$g(t_i) = \frac{(1 - k_i)k_i^{n-1}t_i^n}{\prod_{j \neq i} k_j},$$

$$E[g(t_i)] = \frac{(1 - k_i)k_i^{n-1}}{(n+1)\prod_{j \neq i} k_j}.$$

<number>

So this brings us to variance reduction. The idea of Control Variates, which is a standard technique for Monte Carlo simulation, is to adjust the sampled payoffs for luck. For example, in FPSB, we expect an agent's valuation to correlate positively with its surplus. So we bump up an agent's payoff when it has a low valuation (type) and scale it down when it has a high valuation such that the positive and negative adjustments average out to zero. Then by sampling these adjusted payoffs it will tend to take fewer samples to converge to a good approximation of the true expected payoffs. And for any exogenously determined estimating function $g(\cdot)$ —and I'll explain exogenously in a minute—the average of the adjusted payoffs will always be unbiased and have less than or equal the variance as for unadjusted samples. For FPSB, we can derive a sort of best case control variable, namely, the exact expected payoff for an agent of type t playing strategy k against an arbitrary set of other strategies.

(There are 3 special cases: (1) $g(t_i) = 0$ if $k_i = 0$ and for some $j \neq i$, $k_j > 0$,
 (2) $g(t_i) = 1/2n$ if $k_i = 0 \ \forall i \in \{1, \dots, n\}$,
 (3) otherwise, with z players playing $k = 0$, ignore them and substitute $n - z$ above.)

Control Variates: Estimating $g(t)$

- Bootstrapping a $g()$ – learning a function from types to payoffs from empirical samples
- Semi-automated approach: pick summary statistics and perform linear regression
 - $g(t) = B x(t) + A$
- Unbiased as long as the regression parameters are estimated from a distinct data set (exogenously determined $g()$)

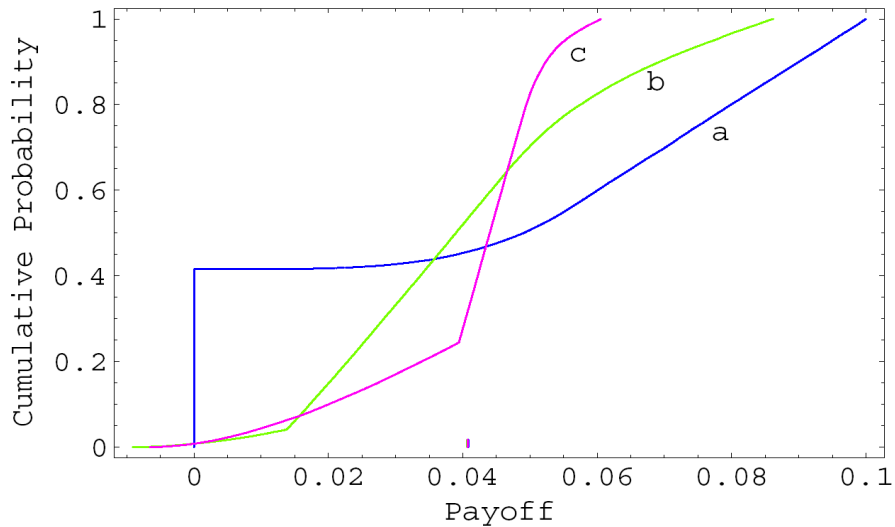
<number>

Of course, in more complicated games, we can't derive anything like that. But instead we can estimate a function using machine learning methods. For FPSB, with one-dimensional types, this is straightforward (for example, we can use linear regression) but in general, with multidimensional types, including other sources of randomness in the game, it may not be obvious how all the random elements influence payoffs. In fact, the dimensionality doesn't have to be very high before it becomes very hard to empirically determine meaningful relationships between types and payoffs. For example, imagine a game involving bidding for many goods with an agent's type being the vector of valuations for each. Depending on the specifics of the game we might expect the sum or the max of an agent's valuations to correlate with payoff. It would take a sophisticated learning algorithm with a lot of data (ie, many game simulations) to rival a simple linear regression from sum or max valuation to payoff. Thus, when we have sufficient domain knowledge—such as knowing that the sum or the max are good summary statistics, we introduce control variates manually.

[In fact, we haven't yet but intend to try introducing control variables of that flavor in the SAA domain. For TAC, we have been using the control variates method for some time, introducing 4 different control variables such as sum of hotel and entertainment premiums, and initial flight quotes.]

<number>

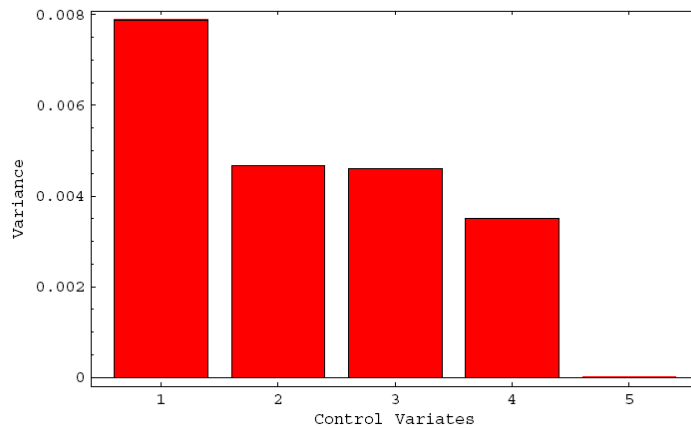
Unadjusted and adjusted Payoffs in FPSB



Returning to our FPSB example, this graph gives a visual sense of what adjustment via control variables can buy us. The blue line is an empirical distribution of unadjusted payoffs for a particular profile. The green line shows adjusted payoffs using a $g()$ determined by linear regression. The pink line is the same but with a separate $g()$ custom fitted for the profile, which is something that's not typically feasible in realistic sized games where we may have only a handful of games per profile. That's plenty to find a generic function from types to payoff estimates but not for tailoring the function for each profile. I don't have the analytically determined control variable included on this graph but...

In the interest of time I'll just flash up the results quickly but these are cdfs of unadjusted and adjusted payoffs for a particular profile of strategies (so the more squished together the cdf the lower the variance).

Average Variance Reduction



- 1. Unadjusted (0)
- 2. Linear regression, profile non-specific
- 3. Linear regression for this profile
- 4. Analytically determined $g()$
- 5. $g()$ = exact expectation (sanity check)

<number>

Here are the average variances for several control variate methods applied to the symmetric equilibrium profile of FPSB4. You can see that in this case it turns out not to matter significantly whether we estimate the $g()$ based on the specific profile or across all profiles.

3. Unadjusted (0)
4. Linear regression, profile non-specific
5. Linear regression for this profile (happened to be about the same)
6. Analytically determined $g()$ for FPSB4
7. $g()$ = exact expectation (sanity check)

Other Variance Reduction Techniques

- Quasi-random Sampling
 - Importance Sampling
 - Stratified Sampling
 - Combined and Adaptive techniques
-
- (Application requires special handling of randomness from Nature)

<number>

There are a some other variance reduction techniques that we haven't employed in any experiments but that promise huge computational savings in empirical payoff estimation. Just to mention them, in case anyone is taking notes on how to apply this methodology to their own monster games, quasi-random sampling and importance sampling are ways to tweak the distribution of types that you sample from to reduce variance in the sampled payoffs. Adaptive techniques combine various methods and adjust the parameters for them automatically as more samples are gathered. Implementations of all of these techniques are available as part of the GNU Scientific Library. Also, I'll refer you to my forthcoming paper for how to deal with Nature when applying these methods.

Player Reduction

- Definition: $\Gamma \downarrow_p$

$$\hat{u}_i(s_1, \dots, s_p) = u_{q \cdot i}(\underbrace{s_1, \dots, s_2, \dots}_{q}, \dots, \underbrace{s_p, \dots}_{q})$$

- Theorem: unique symmetric equilibrium of $\text{FPSB}n \downarrow_p$ is

$$\frac{n(p-1)}{p+n(p-1)}$$

<number>

The next method for taming monster games I call player reduction and the idea is very simple: we can approximate an n-player game with one of, say, n/2 players by an n/2 player game where each player gets to pick a strategy for 2 players to play in the original game. Since game size is exponential in the number of players, this can drastically cut an impossibly large game down to size, as we'll see in the case of the Trading Agent Competition.

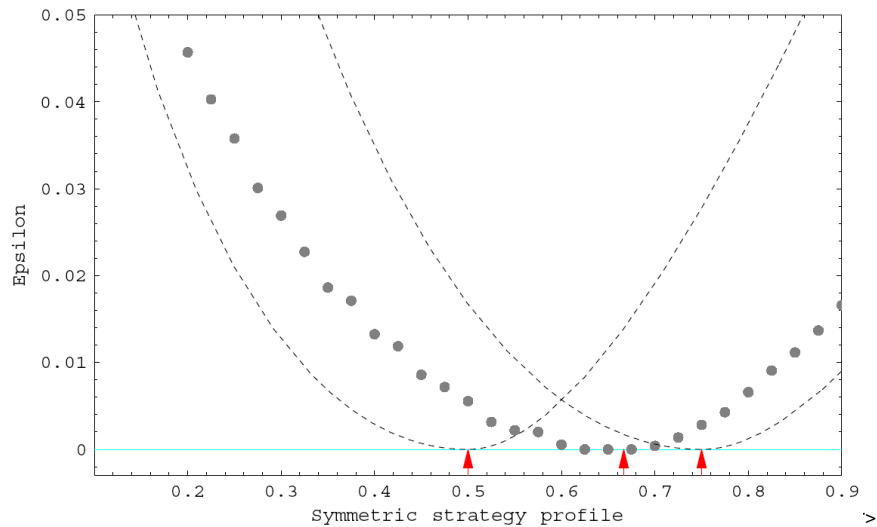
n

p = n/2

q = 2 = n/p

[player reduction description, yoking strategies, take a pq player game and define the p-player game where each player picks one strategy for q players to play]

Epsilons for Symmetric Profiles in FPSB2, FPSB4₂, FPSB4



Now the question is whether the reduced game bears any similarity to the original game. Of course in the worst case it won't – we can make up a pathological example where the payoffs vary in any erratic way you can imagine. But our hypothesis was that for many natural games the degradation would be graceful.

That's borne out for FPSB ... [DESCRIBE]

FPSB2 -> $\frac{1}{2}$

FPSB4r2 -> $\frac{2}{3}$

FPSB4 -> $\frac{3}{4}$

Theoretical Results for FPSB

- Theorem: For all $n > p \geq 1$

$$eq(\text{FPSB}p) < eq(\text{FPSB}n \downarrow_p) < eq(\text{FPSB}n) \quad \text{and}$$

$$0 = \varepsilon(eq(\text{FPSB}n)) < \varepsilon(eq(\text{FPSB}n \downarrow_p)) < \varepsilon(eq(\text{FPSB}p))$$

(reducing to p players always outperforms the p player version)

- Theorem: For all $n > p > q \geq 1$

$$eq(\text{FPSB}n \downarrow_q) < eq(\text{FPSB}n \downarrow_p) < eq(\text{FPSB}n) \quad \text{and}$$

$$0 = \varepsilon(eq(\text{FPSB}n)) < \varepsilon(eq(\text{FPSB}n \downarrow_p)) < \varepsilon(eq(\text{FPSB}n \downarrow_q))$$

(solution quality degrades monotonically with more severe player reduction)
<number>

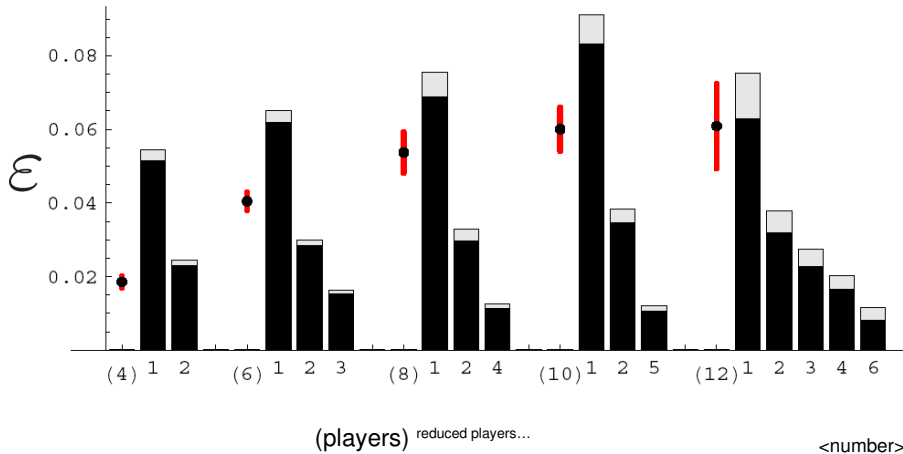
And in fact we proved that for FPSB, that generalizes to any number of players.

If you have an n-player game but computationally you can only handle p players then you're better off with the p-player reduction than the actual p-player version, both in terms of absolute closeness to the n-player Nash equilibrium and in terms of the epsilon metric.

And for FPSB, solution quality degrades monotonically with more severe player reduction.

[Theorem 3.12 and 3.13]

Local-Effect Games



So those are reassuring results for FPSB, but of course we don't actually need to approximate FPSBn since we already know the solution. So to further evaluate the quality of reduced-game approximations, we turned to other natural games of potential interest. Local-effect games fall under the category of congestion games, for example deciding what roads to take when you have to trade off taking the most direct route with the possibility that too many other agents will choose the same and the route will be slower.

Just to summarize the conclusion: player reduction does well at approximating games in this class as well and again we find that the solution quality degrades gracefully with the degree of reduction.

I'll discuss player reduction in the context of the trading agent competition shortly, where it was critical in getting any kind of strategic handle on that monster game.

Figure 3.6: Local-effect games with 4, 6, 8, 10, and 12 players. Each group of bars shows the average epsilon

for equilibria of reductions of the given game at increasing fidelity. The number of players in the full game

is shown in parentheses, with the number in reduced games under each bar. The bars extend upward to

indicate a 95% confidence upper bound on eps. To the left of each group is shown the eps (with 95% confidence

interval) of the social optimum of the full game.

Analyzing Empirical Games

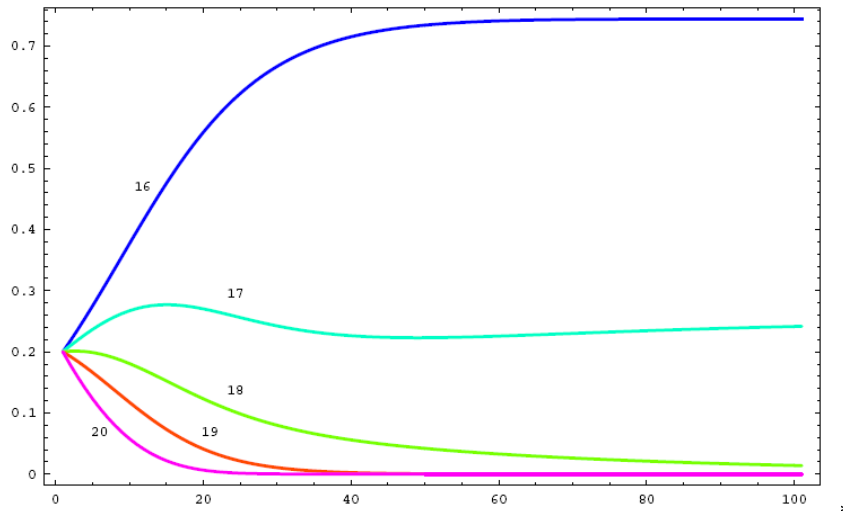
- Gambit
- Replicator Dynamics
- Function Minimization (Amoeba)

<number>

Everything so far has been about estimating an empirical game. Once we have that, we have a standard normal form game with a finite payoff matrix and we can apply any standard game solving techniques. Just to mention the ones we've used: Gambit is the state of the art solver for finite games. But since it doesn't exploit symmetry we've used 2 other methods that do, namely replicator dynamics and function minimization.

Replicator dynamics bears special mention. Although the idea is not new to me and my colleagues (in fact, it originated as a model for how animals evolve toward Nash equilibrium strategies) we are the first to our knowledge to apply it as a solution technique for large games.

Example of Replicator Dynamics



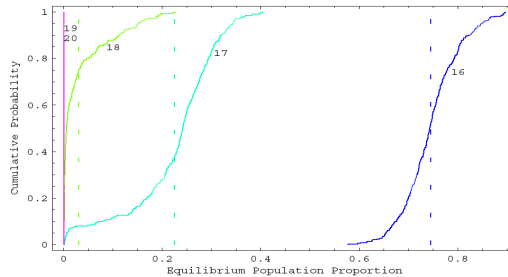
The idea is simple...

Start with a population of strategies with each represented equally in the population ($1/5$ each here). Then grab 5 of them from the population, play them against each other, and adjust their numbers in the population in proportion to how well they did. And repeat for as many generations as necessary till you reach a fixed point, which corresponds to a mixed-strategy Nash equilibrium.

Figure 3.7: Replicator dynamics for an SAA game with 5 players and 5 strategies $\{16, \dots, 20\}$ evolving in about 100 generations to a symmetric mixed equilibrium of all agents playing strategy 16 with probability 0.754 and 17 otherwise.

Sensitivity Analysis

- Distributions over payoff matrices
- Distributions over functions of the payoff matrix
 - Nash equilibrium
 - Epsilon for a Nash *candidate*
- Confidence bounds on mixture probabilities
- Confidence bounds on epsilon

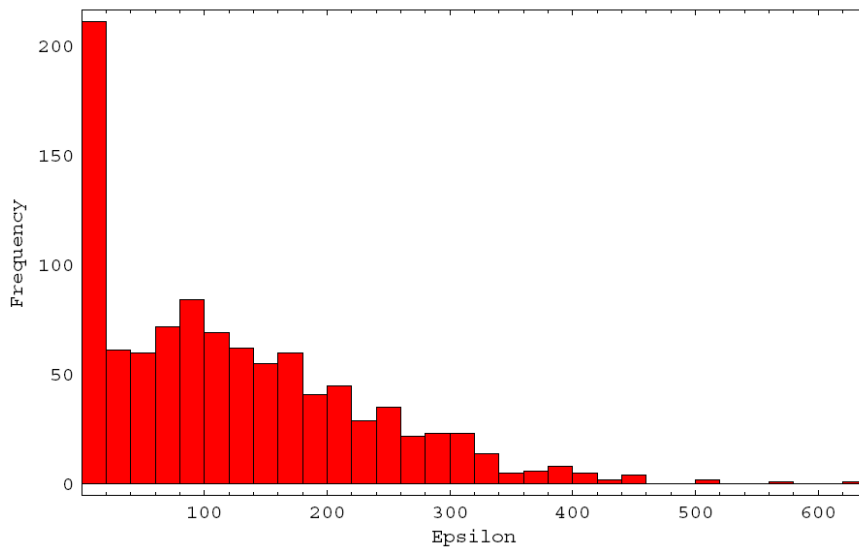


...finally, the last phase of our monster game taming methodology is assessing solution quality with respect to the underlying game of interest, or at least with respect to the exact restricted game, after reducing the strategy space and possibly the number of players. To do this, we first need to estimate a distribution representing our belief over the space of possible payoff matrices, which we do based on our sampling data (though I won't go into the details of how we do that).

One method of sensitivity analysis, then, is to see if enough samples from the payoff matrix distribution all yield the same equilibrium, in which case we can conclude that our results are robust to sampling noise. One way to operationalize that is to repeatedly sample payoff matrices, compute one or all symmetric equilibria, and observe the empirical distributions over mixture probabilities. [EXAMPLE] (we see that the equilibrium derived by replicator dynamics from the previous slide actually has a fair bit of potential error – confidence intervals on the mixture probabilities)

[We do this by making some conservative independence assumptions and gathering sample statistics for individual cells in the payoff matrix. We exploit the central limit theorem which tells us that as the number of samples grows, the distribution of the sample mean approaches normal. It is this normal distribution that we take as our belief over the individual expected payoffs in the payoff matrix. We can now get sample payoff matrices by sampling the individual cells.]

Sensitivity Analysis



An alternative is to do the same for the epsilon metric. Here's an example from a different game in which we can say with 15% probability (that's how much of the probability mass is at zero) that this profile is an equilibrium.

[and compute, for a candidate equilibrium (like the equilibrium of the estimated empirical game) an empirical distribution for epsilon, just like we did for mixture probabilities. If a fraction p of the probability mass is at zero then we conclude with confidence p that we have found an actual equilibrium of the underlying restricted game. In the best case, p is near one and the conclusion is definite. If not, we can at least give probabilistic bounds on epsilon and quantify the degree to which the candidate equilibrium approximates an equilibrium to the true game.]

Figure 3.10: eps-sensitivity analysis for a 2-player reduced TAC game with 35 strategies showing the empirical

pdf (histogram) for eps. The expected eps is 123 and the probability that the candidate is an actual

equilibrium (eps = 0) is 15%. More probability mass near zero means a more robust profile. The partial

payoff matrix was estimated from 14,000 samples spread (non-uniformly) over the 630 profiles, adjusted

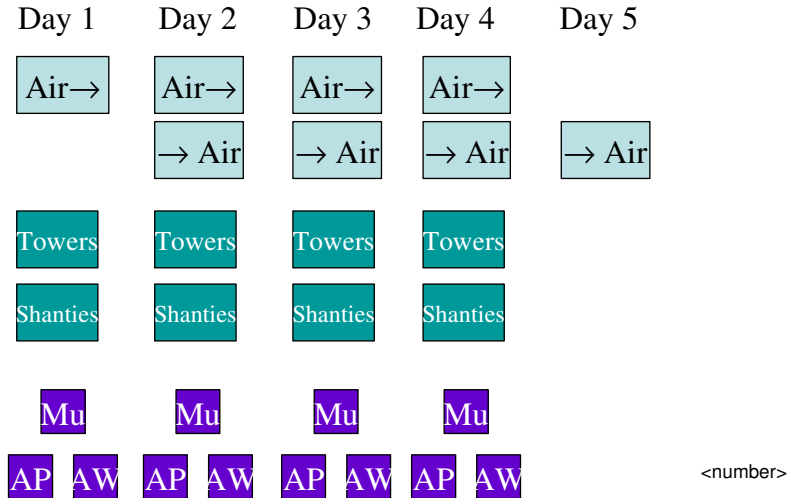
with control variates.

Conclusion: Taming Monster Games

- Restrict strategy space
- Reduce number of players
- Simulate game outcomes
 - Adjust type distributions or adjust sampled payoffs with control variables to reduce variance
- Analyze empirical game
- Assess solutions wrt underlying game

<number>

(Simultaneous Auctions in) TAC Travel



So, on to the real applications of monster game taming. We created the Trading Agent Competition Travel shopping in 2000 and it has been growing ever since. Starting in 2002 the Swedish Institute of Computer Science took over running and we've been competing in it since then. The game pits 8 travel agents against each other, all trying to put together travel packages for their hypothetical clients. I won't get into the intricacies of the game except to say that there are 28 simultaneous auctions of different types that the agents have to participate in to buy flights, hotels, and entertainment tickets. One of the key strategic issues is the strong complementarity between hotels. If you have a client staying in the Towers (that's the hypothetically fancy hotel) on day 1 and they don't leave till day 3 then that first room is useless to you unless you also get a room in the same hotel on day 2. Actually, it's a bit messier than that since you may be able shuffle your clients around, shorten their trips if you haven't bought their flights yet, etc. But fundamentally, we have a problem of bidding for complementary goods in simultaneous ascending auctions. The other domain we study distills that problem out specifically, making for a much simpler game, though still a multistage game for which we need our empirical game methodology to make any headway. That game is called SAA for Simultaneous Ascending Auctions.

“Walverine”



wolverine



Léon Walras



<number>

The name of our agent is Walverine, from the Michigan mascot and the 19th century French economist, Walras, because the first foundational idea of our agent was to predict market prices using a simple market model, namely Walrasian equilibrium.

(The reason price prediction is important is because of the exposure problem. Suppose I don't care much about the nice vs cheap hotel but the current prices are all low so I start bidding for the nice rooms for all the nights of my stay (and remember, I can't switch hotels mid-trip). Now imagine of the nice rooms starts to skyrocket. I can drop out of the bidding of course but if I'm already winning a nice room on the first night then I'm stuck. I'd have to just eat it and pay for one nice room that I can't use. Whereas if I'd predicted which rooms would get expensive I could've rearranged my trip when the flights where still cheap, or just bid for the cheap hotel from the start.)

TAC Price Prediction

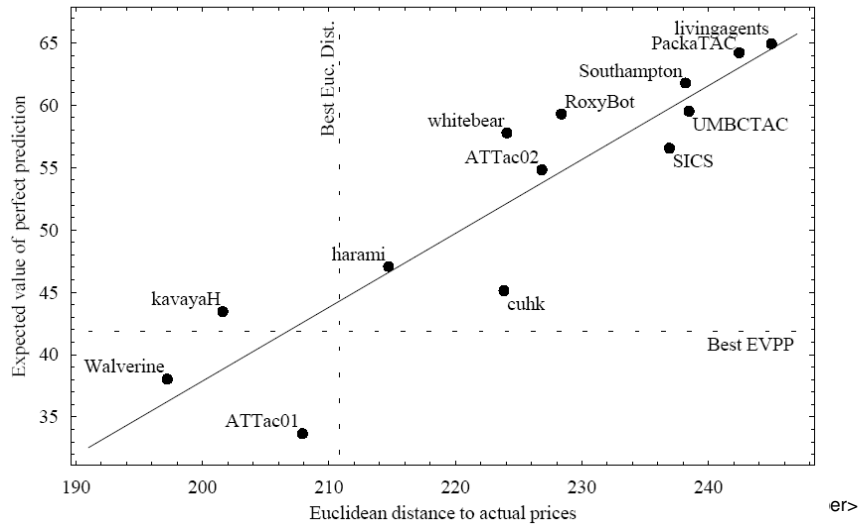
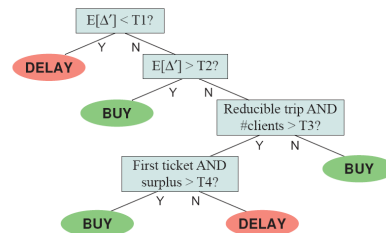


Figure 4.2: Prediction quality for thirteen TAC-02 agents. Dashed lines delimit the accuracy achievable with constant predictions (independent of flight prices and own client preferences): “best Euclidean distance” and “best EVPP” for the two respective measures. The diagonal line is a least-squares fit to the points. Observe that the origin of this graph is at (190,32).

Parameterized TAC Agent

- Walrasian Price Predictor with...
- Variations on hotel bid shading
- Different entertainment trading strategies
- Parameterized decision process for flight purchase timing



But how this relates to our monster game methodology is that we started with the baseline strategy of our Walrasian price predictor, and just like in the FPSB example, introduced parameters to generalize it...

Searching for Walverine (among 40 candidate strategies)

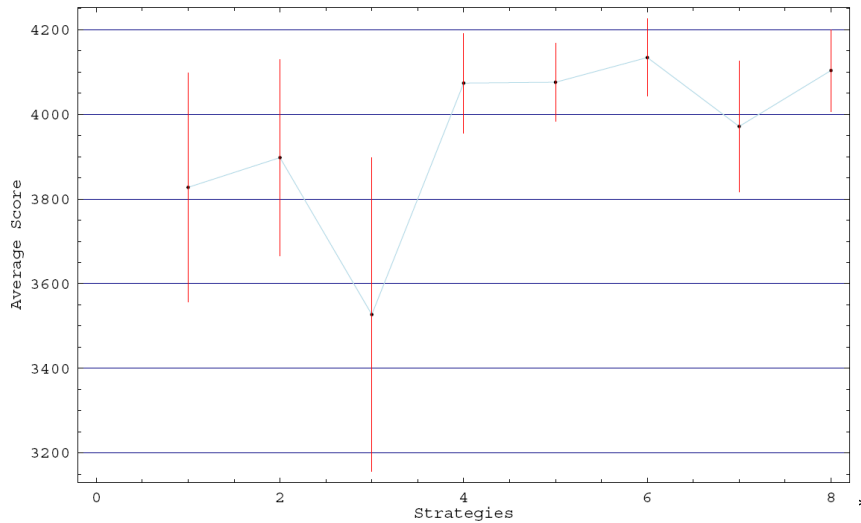
p	Profiles			Samples/Profile	
	total	evaluated	%	min	mean
8	> 314M	2114	0	12	22.3
4	123,410	2114	1.7	12	22.3
2	820	586	71.5	15	31.7
1	40	40	100	25	86.5

Profiles Evaluated in Reduced TAC Games

<number>

Table 6.4: Profiles evaluated in reduced TAC games (TAC p).

Performance of 8 Walverine Variants



I'll skip the details but based on our empirical game theoretic analysis of playing variants of Walverine against each other we ended up with several candidate strategies which we then tested in real tournament conditions, in the seeding and qualifying rounds, before settling on the final tournament version of Walverine...

Figure 6.9: Performance of eight Walverine variants, {3, 4, 16, 17, 35, 37, 39, 40}, in the TAC-05 seeding rounds, based on 507 games.

Walverine in Third...



WELCOME TO TRADING AGENT COMPETITION

Competitive Benchmarking for The Trading Agent Community

[LOGIN](#) | [SEARCH](#) | [FAQ](#) | [LEGAL NOTICES](#)

[TAC 2005 Finals](#) | [TAC 2005 Semi-Finals](#) | [TAC 2005 Seeding](#) | [TAC 2005 Qualifying](#)

Score tables for TAC 2005 Finals - 3rd August

TAC Classic

Position	Agent	Average Score	Games Played	Zero Games
1	Mertacor (tac1 , tac2)	4126.49	80	0
2	whitebear05 (tac1 , tac2)	4105.68	80	0
3	Walverine (tac1 , tac2)	4058.90	80	2
4	Dolphin (tac1 , tac2)	4022.79	80	0
5	SICS02 (tac1 , tac2)	3972.29	80	0
6	LearnAgents (tac1 , tac2)	3899.24	80	0
7	e-Agent (tac1 , tac2)	3451.25	80	0
8	RoxyBot (tac1 , tac2)	3167.64	80	10

The scores have been combined from the TAC 2005 finals at [tac1.sics.se](#) and [tac2.sics.se](#).

TAC SCM

GENERAL

- [Home](#)
- [About TAC](#)
- [News](#)
- [Press](#)
- [Calendar](#)
- [Contact Info](#)

JOIN NEWS LIST

TAC 2005

- [Info & Call](#)
- [Participants](#)
- [TADA Workshop](#)
- [Results](#)

PREVIOUS RESULTS

- [TAC 2004](#)
- [TAC 2003](#)
- [TAC 2002](#)

TAC COMMUNITY

- [Research Groups](#)
- [Educating using TAC](#)
- [TAC Policies](#)

And here are the official results.

Walverine Rules (p=.17)

Recalculated Scores for TAC 2005 Finals

Position	Agent	Average Score	Games Played	Zero Games
1	Walverine (tac1 , tac2)	4157.10	58	0
2	RoxyBot (tac1 , tac2)	4066.94	58	0
3	Mertacor (tac1 , tac2)	4063.17	58	0
4	whitebear05 (tac1 , tac2)	4001.89	58	0
5	Dolphin (tac1 , tac2)	3993.31	58	0
6	SICS02 (tac1 , tac2)	3904.70	58	0
7	LearnAgents (tac1 , tac2)	3785.36	58	0
8	e-Agent (tac1 , tac2)	3369.99	58	0

The scores have been combined from the TAC 2005 finals at [tac1.sics.se](#) (games 30563-30591) and [tac2.sics.se](#) (games 12108-12136).

This is a recalculation of the TAC 2005 final scores where some problem games have been removed (RoxyBot was running two agents on tac1.sics.se and no agent at tac2.sics.se during the first games).

And I just have to mention that although officially Walverine came in third, the organizers also published these unofficial results with some tainted games removed (including 2 in which we had a network outage) and so unofficially Walverine kicked butt (at the p=.17 significance level).

Strategy Generation Summary

- First algorithm to compute best-response strategies in a broad class of infinite games of incomplete information
- Empirical game methodology for applying game-theoretic analysis to much larger games than previously possible
- Theoretical and experimental evidence of the efficacy of our methodology
- A price-prediction approach to strategy generation in simultaneous auctions for complementary goods
- Application of the above methods to find good strategies in complex games (TAC and SAA)
- **Future Work:** new domains (multiattribute auctions, sponsored search auctions), computational mechanism design

<number>

So, to conclude, I've shown you my best response solver for a broad class of infinite games, our empirical game methodology for taming monster games, demonstrated it on simple games like FPSB, and showed how it helped us (unofficially) win the Trading Agent Competition in 2005.

In future work, I plan to apply these techniques to new domains like multiattribute auctions and other complex market mechanisms including potentially sponsored search auctions.

And I'm especially eager to apply my strategy generation techniques to computational mechanism design problems. The reason that strategy generation is key to that problem is that, say Yahoo is considering various changes to the sponsored search auction. The only way to evaluate potential changes to the mechanism is to predict what advertisers will actually do in response, and that means strategy generation.

So whether we model these games as one-shot infinite games that are captured by my analytic techniques or as more complex games for which my empirical methodology is needed, strategy generation is critical for agents and mechanism designers alike.

Click to add title

- Click to add an outline

<number>

Backup Slides

Click to add text

<number>

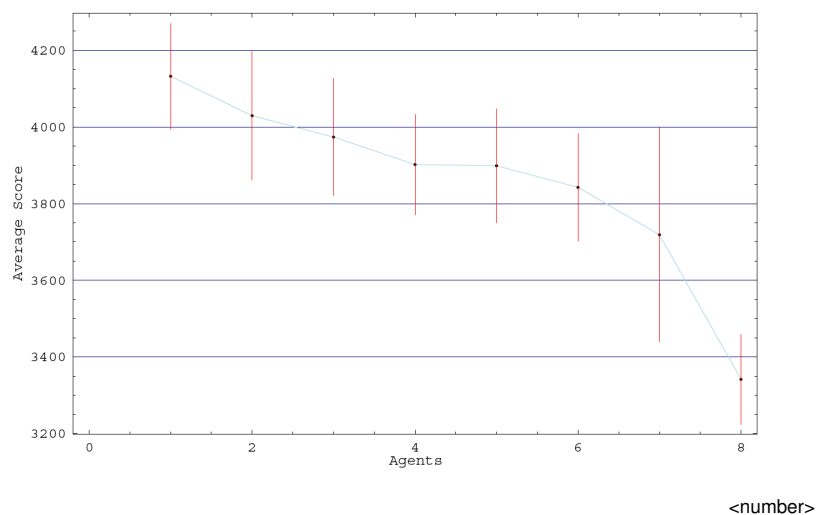
Strategy Metrics

Strategy	Count	Max	Sum
17	24	0.499	5.39
4	19	0.729	4.86
21	17	0.501	4.68
16	18	0.892	3.77
23	14	0.542	3.34
6	16	0.699	3.25
9	16	0.367	3.09
5	23	0.247	2.63
24	17	0.232	2.34
35	15	0.641	2.04
40	20	0.180	1.95
3	8	0.401	1.70
34	7	0.307	1.05
7	6	0.099	0.37
38	5	0.091	0.35
39	3	0.126	0.18

<number>

Table 6.6: For all strategies appearing in an unrefuted equilibrium of a clique in TAC 2, the number of equilibria, the maximum mixture probability, and the sum of all mixture probabilities across equilibria.

Comparison of TAC-05 Scores

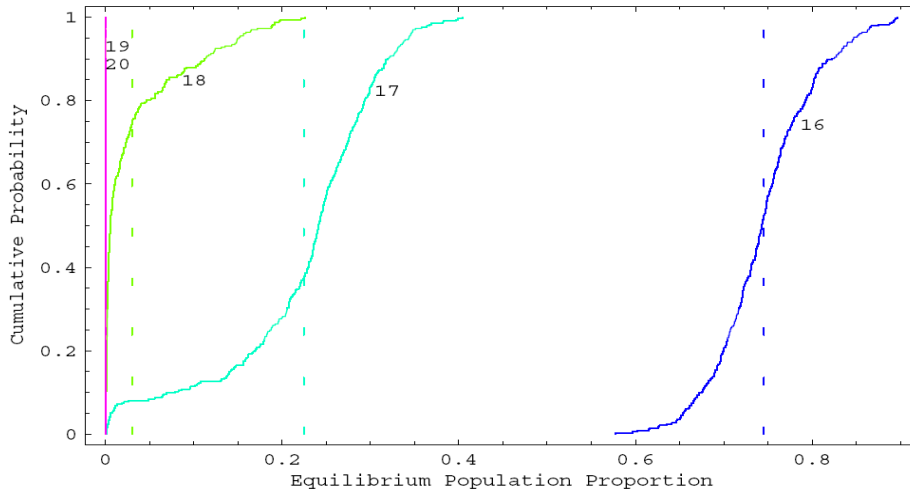


Applying 2004-derived control variates achieves 17.5% variance reduction in 2005.

MEAN DIFF TESTS:

	2	3	4	5	6	7	8
1	0.172	0.063	0.008	0.012	0.002	0.005	0.
2	-	0.312	0.116	0.122	0.045	0.03	0.
3	-	-	0.237	0.241	0.104	0.056	0.
4	-	-	-	0.488	0.268	0.119	0.
5	-	-	-	-	0.291	0.129	0.
6	-	-	-	-	-	0.215	0.
7	-	-	-	-	-	-	-
0.007							

Sensitivity Analysis 1



[describe] Unfortunately, we often don't have the simulation time needed to get the sample variances low enough to be able to say anything at all about the likely ranges of mixture probabilities in equilibrium. (though whenever a pure equilibrium is identified it tends to be far more robust, less sensitive, and this method can be useful to confirm equilibria even with relatively few samples) So as an alternative sensitivity assessment, we use our epsilon metric...

Price Prediction for Complementary Goods in Simultaneous Auctions

- Point vs. distribution prediction
- Self-confirming prediction
- Walrasian price equilibrium prediction
 - Prices such that supply meets demand
- Both can be found (at least approximately) by an iterative process

<number>

In chapter 4 I describe those 2 games in detail and lay out our approach to applying the first step of the empirical game methodology, namely generating candidate strategies. That approach for both TAC and SAA is price prediction. There are 2 basic kinds of price prediction: point predictions of final prices and predicting by coming up with a probability distribution representing your belief about the final prices. One key method of predicting prices is self-confirming prediction: the idea there is to define a prediction based strategy parameterized by the prediction vector or prediction distribution and then find a prediction such that if everyone plays that prediction strategy the prediction will turn out to be right.

Another key method of predicting prices is Walrasian price equilibrium, and this is how our TAC agent, Walverine, got its name.

A Walrasian price equilibrium is a set of prices such that supply meets demand. Of course, it's a bit trickier than that (for one thing, price equilibria need not exist in these games, though they seem to at least approximately in the examples we've studied) but also, we don't actually know the other agents' demands since those are determined by their types, ie, their valuations for possible bundles of goods. But we do know the distribution...

Performance of Self-Confirming Prediction in SAA Games

$SAA_{\lambda \sim *}(m, n)$	$\varepsilon\% (PP(F^{SC}))$	$\bar{\varepsilon}\%$	$Pr(\varepsilon = 0)$	$Pr(PP(F^{SC}))$
$E(3, 3)$	0	0	1.00	1.00
$E(3, 5)$	0	.09	.600	.996
$E(3, 8)$.83	.85	0	—
$E(5, 3)$	0	0	1.00	.999
$E(5, 5)$	0	.01	.900	.998
$E(5, 8)$.60	.64	0	—
$E(7, 3)$	0	.06	.667	.992
$E(7, 6)$.04	.10	.567	.549
$U(3, 3)$	1.24	1.26	0	.725
$U(3, 5)$	0	0	1.00	1.00
$U(3, 8)$.56	.53	0	—
$U(5, 3)$	1.35	1.35	0	.809
$U(5, 8)$	1.59	1.62	0	—
$U(7, 3)$.81	.84	0	.942
$U(7, 6)$.52	.52	0	.929
$U(7, 8)$	4.98	4.94	0	—

<number>

Table 5.3: Performance of PP(FSC) as a candidate symmetric equilibrium for various SAA, U and SAA, E environments.

The Trading Agent Competition

- 12 minute games, 8 agents competing per game
- Agents perform as travel agents, purchasing travel goods for clients at auction
- Each travel agent given clients requests, defining objective function. Net value is objective minus expenditure
- Assemble trip for each client, comprising flight, hotel, and entertainment
- Goods are interdependent, each presents interesting issues

<number>

Shading vs. Non-Shading Walverine

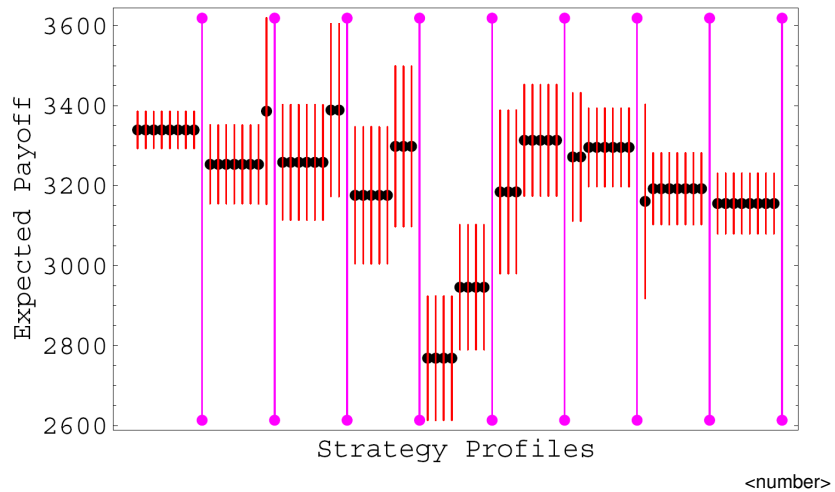


Figure 6.1: Empirical payoff matrix for shading vs. non-shading Walverine, based on roughly 100 games per profile. The profile of all shaders is on the left and all non-shaders on the right.

Symmetric Strategies in TAC₁

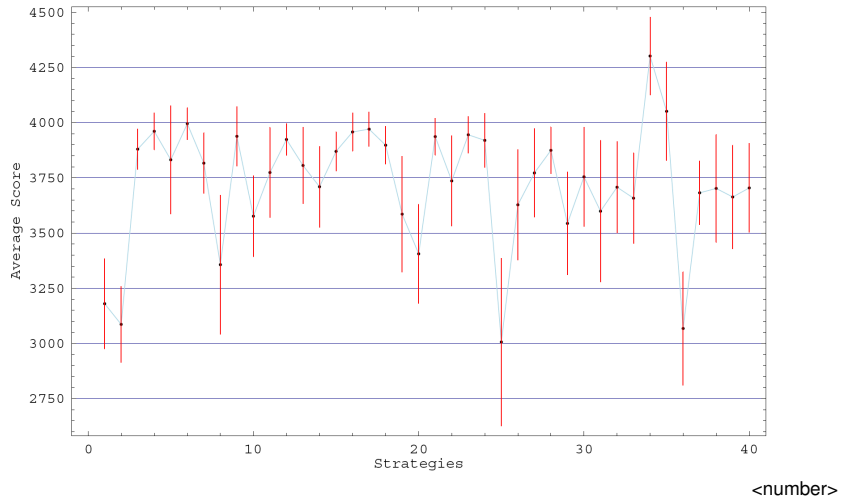


Figure 6.4: Average payoffs for (symmetric) strategy profiles in TAC₁. Error bars delimit 95% confidence intervals.

Prisoners' Dilemma

	17 (D)	34 (C)
17 (D)	3971	4377
34 (C)	3907	4302

<number>

Table 6.5: The TAC game restricted to strategies 17 and 34 constitutes a prisoner’s dilemma where 34 is

“cooperate” and 17 is “defect”. Note that the temptation payoff > reward payoff > punishment payoff >

sucker payoff. Additionally, the reward payoff exceeds the average of the temptation and sucker payoffs.

Why “Strategy Generation”? (SG)

- Infinite Games: SG is meant in the sense of a game solver (generating equilibrium strategies from a game description)
- Monster Games: SG refers to the process of generating a set of candidate strategies as well as choosing among them
- Price Prediction: families of strategies from which we generate prediction specific strategies
- TAC/SAA: applying empirical game methodology to establish good strategies in two market games

<number>

TAC-05 (if not for meddling kids)

Agent	Raw Score	Adjusted Score	95% C.I.
Walverine	4157.1	4132.42	± 138.4
RoxyBot	4066.94	4029.53	± 167.3
Mertacor	4063.17	3973.9	± 152.3
whitebear05	4001.89	3902.01	± 130.4
Dolphin	3993.31	3898.95	± 148.5
SICS02	3904.7	3842.61	± 140.6
LearnAgents	3785.36	3718.81	± 280.0
e-Agent	3369.99	3341.52	± 117.2

<number>

Price Prediction Can Help

Agent	{1}	{2}	{3}	{1, 2}	{1, 3}	{2, 3}	{1, 2, 3}
1	0	0	0	0	0	0	15
2	8	6	5	8	8	6	8
3	10	8	6	10	10	8	10

<number>

4.2: Agent valuation functions for a problem illustrating the value of price prediction over SB and sunk-aware agents.

Control Variates for FPSB4

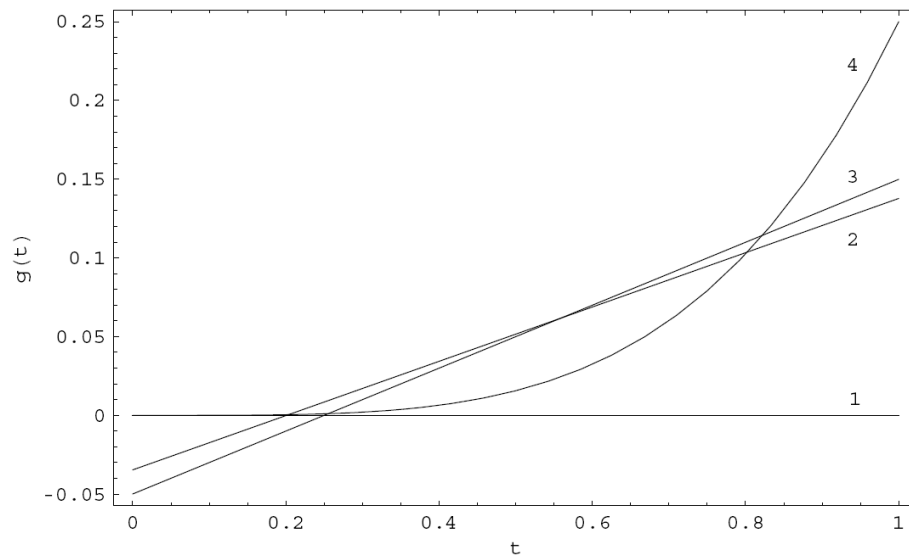
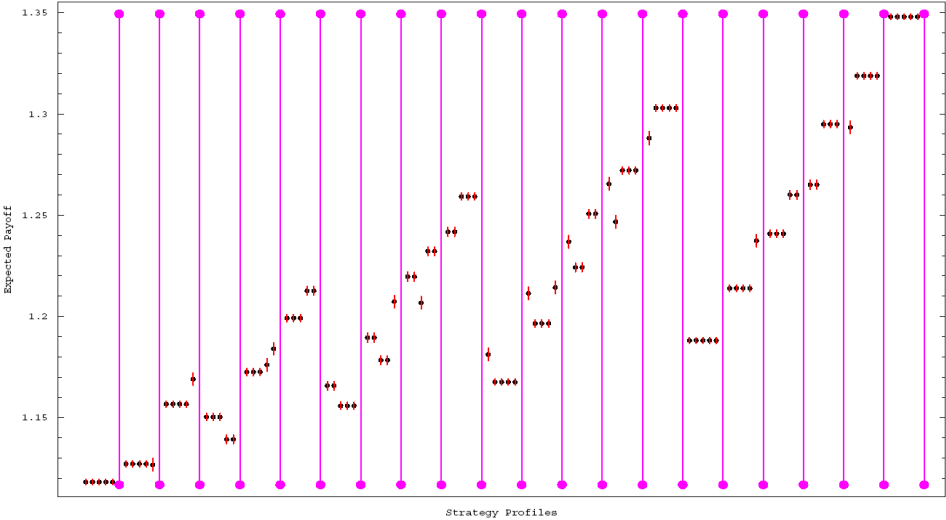


Figure 3.3: Four control variates, $g(t)$, for FPSB4.

Empirical Payoff Matrix



Finite Game Approximations

- Finite game solvers:
 - Gambit
 - Gala
 - Gametracer
- Why not discretize?
 - Introduces qualitative differences
 - Computationally intractable

<number>

Price Prediction Strategies for Market-Based Scheduling

Jeffrey K. MacKie-Mason
Anna Osepayshvili
Daniel M. Reeves
Michael P. Wellman

University of Michigan

<number>

[Introduction] First, a quick caveat: scheduling is the domain that we've applied our approach to but we're not making claims about solving scheduling problems and in particular, we don't claim that *markets* are a good mechanism for solving scheduling problems. What we are interested in is price prediction strategies in the domain of market-based scheduling.

[Clarify about what the mechanism is really trying to do: social welfare? Feasible allocation?]

[say upfront that this question of strategy for SAA is an open problem in general and that the results are not specific to *this* scheduling problem.]

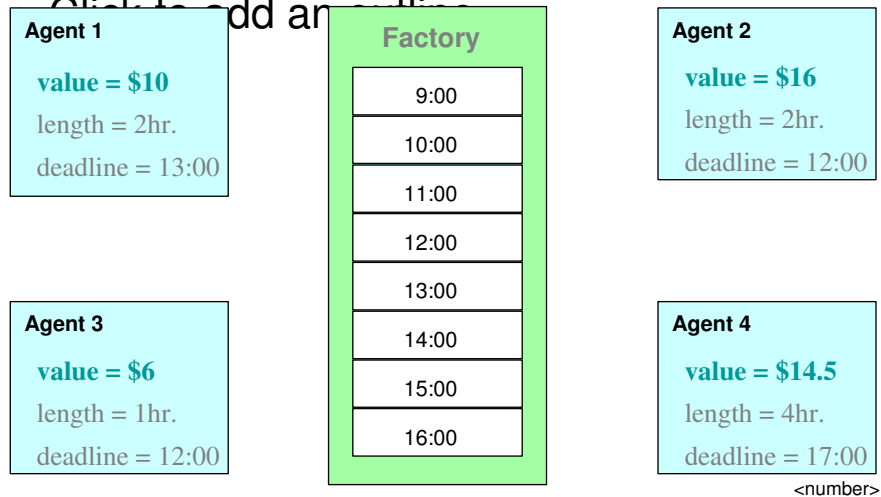
[start with disclaimer about how we are not interested in scheduling (more diplomatic way to say that) and that this is about finding good strategies for a particular (not necessarily good) mechanism for solving the scheduling problem. In other words, we are not claiming that markets are the right way to solve scheduling problems. Our result is that price prediction is a good strategic approach to SAA. Price prediction is not just for scheduling, but for any domain with SAA. There is no known better strategy.]

[motivation for SAA: ebay and auctions for time slots for TV ads]

[when talking about the exposure problem maybe mention how in a single sotheby's auction there's no exposure problem because you just stop bidding when the price gets too high. Very simple.]

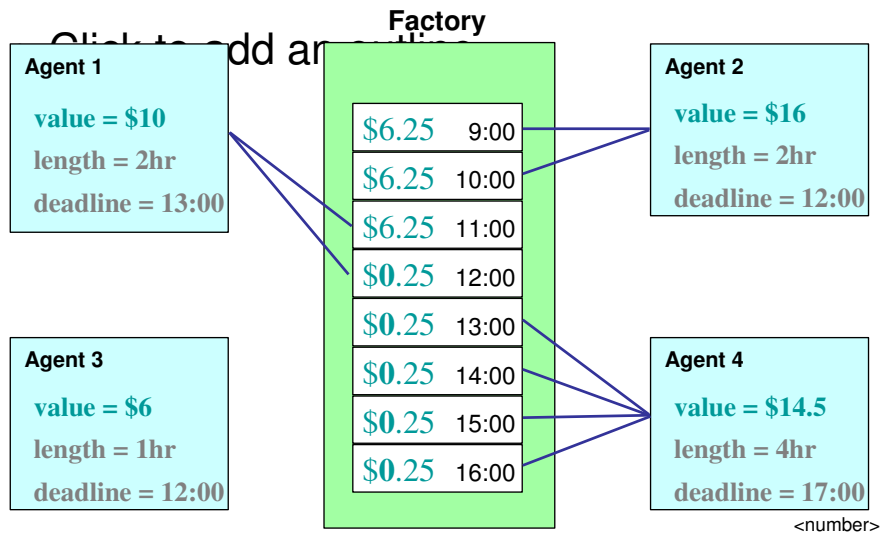
<number>

Factory Scheduling Example



Here's the basic scheduling problem we're interested in. Consider a factory with several time slots available. A set of agents have valuations for different combinations of slots, defined in terms of number of slots needed to complete their job and values for completing it depending on when the latest slot is finished by. The example here is the special single deadline case where every agent has one deadline and one value, with implicitly zero value for not completing the job by that one deadline. The problem is to compute an allocation of slots to agents. AI people might naturally be inclined to encode this kind of problem as a CSP. Indeed, that would be quite straightforward, as would the solution.

Schedule Price Equilibrium



But what if the agents are autonomous and their preferences are private? One way to determine an allocation while respecting their autonomy is to find a set of prices that induces an allocation. In this example, the prices are such that every agent wants a certain bundle of slots and every slot is wanted by at most one agent. Prices satisfying that property are said to be in equilibrium and whenever a price equilibrium exists the corresponding allocation maximizes social welfare. Unfortunately, price equilibria don't always exist. Nonetheless, market mechanisms – that is, price-based mechanisms are ubiquitous for resource allocation problems such as this scheduling domain.

Claimed that prices isolate the factors an agent needs to make a decision. We say that the prices are in *equilibrium* if the decisions that the agents make given the prices are consistent with each other.

[Explain why this system is in equilibrium.]

Equilibrium is important because it is what we need to *implement* the allocation in a decentralized manner. Since the agents are autonomous, we need to reconcile their authority with the requirement that the resulting allocation is feasible.

Also point out that allocation from a price equilibrium is not arbitrary, but tends to support the most efficient allocations (more later). In example, only alternative would be to substitute agent 3's job, but the price system shows clearly why this would lead to less total value.

The Market Mechanism

- Agent Preferences
 - Job length
 - Deadline values
- **Simultaneous Ascending Auctions**
 - One auction per time slot
 - Price quotes announced after each round
 - Auctions clear when all are quiescent

<number>

The particular market mechanism we consider is Simultaneous Ascending Auctions. The mechanism (ie, game) that agents are faced with is defined in terms of a distribution from which agent preferences drawn. (preferences being job lengths and valuations for finishing at different times) .. As well as a set of auction rules, namely, that agents place bids in each auction at every round, the auctioneer publishes price quotes which are the current winning bids, and then the next round starts. The mechanism ends when it's quiescent; in other words, no agent chooses to place a new bid in any auction.

Now suppose that an agent is faced with such a market; how should it behave?

Exposure Problem

- Balance benefit of acquiring enough slots with risk of buying unusable slots
- Price prediction can mitigate the exposure problem
- Knowing the eventual price of a slot means you can avoid committing to it

Name	Job Length (λ)	$v(1)$	$v(2)$	$v(3)$	
Agent 1	3	—	—	15	
Agent 2	1	8	6	4	
Agent 3	1	10	8	6	umber>

What makes this problem hard is the complementarities between slots. That is, getting any one slot only helps me if I get enough other slots to complete my job.

The complementarities (along with the fact that the mechanism doesn't allow agents to repudiate bids) are what cause the exposure problem, which is the problem of bidding for a slot and risking that you'll win it and be stuck with it without winning the other slots you need.

I should also mention that there are other market mechanisms – such as the combinatorial auction – that don't have the exposure problem and a combinatorial auction is a great thing to use when possible but it won't always be possible. For various reasons, Simultaneous Ascending Auctions are a fact of life and that's the mechanism we're interested in.

One way to mitigate the exposure problem is to predict where prices are heading.

We hypothesized that this would help by allowing an agent to avoid committing to slots that it can predict will become too expensive.


For example, if Agent 1 knew enough about Agent 2 and 3's preferences to see that there's no way it can win its 3 slots for less than its value of 15 then it can avoid bidding altogether

[also mention that a combinatorial mechanism would solve this problem but, 1, it's complex in terms of computation and communication and, 2, SAA are a fact of life.]

Straightforward Bidding (SB)

No attempt to anticipate other agents' strategies:

- Perceived prices
- Best bundle
 - maximizes surplus at perceived prices
 - assumes it will win the whole bundle

Slots	1	2	3	Surplus = Value - Cost
Current prices (CP)	\$10	\$9	\$5	
Slots the agent is winning	yes	---	---	
Perceived prices 	\$10	\$10	\$6	
Job length	2 slots			
Deadline Values	---	\$35	\$25	
Bundle {1,2}	yes	yes	---	35 - 20 = \$10
Bundle {1,3}	yes	---	yes	25 - 16 = \$9
Bundle {2,3}	---	yes	yes	25 - 10 = \$15

We'll start by describing a basic myopic strategy we call Straightforward Bidding.

Consider the case of 3 slots which have so far been bid up to 10, 9, and 5 dollars. Our agent is the current winner of slot 1.

We assume a minimum bid increment of 1 dollar, so the agent decides its "perceived prices" for the slots as the current price quote for any slot its winning and the price quote plus the bid increment for slots it's not winning. The idea is that these are the minimum prices it can expect to pay for the slots.

The agent uses its perceived prices to determine what slots to bid on. [CLICK]

It needs 2 slots for its jobs so the possible bundles are 12, 13, and 23, plus implicitly the empty bundle which always has value and cost of 0.

So given the perceived prices and our deadline values, we can compute surpluses for the possible bundles and pick the one with highest surplus and place bids on the slots in that bundle that we aren't already winning.

[make sure not to say "these", rather, eg, "perceived prices" or "bundle 1"]

Modification Of SB

Estimate final prices:

- Perceived prices
- Information on market prices
- Best bundle
 - maximizes surplus at estimated final prices
 - assumes it will win the whole bundle

Slots	1	2	3	Surplus = Value - Cost	
Current prices (CP)	\$10	\$9	\$5		
Slots the agent is winning	yes	---	---		
Perceived prices	\$10	\$10	\$6		
Vector of predicted prices (π)					
Adjusted prediction = $\max\{\text{perc}, \pi\}$					
Job length	2 slots				
Deadline values	---	\$35	\$25		
Bundle {1,2}	yes	yes	---		35 - ?
Bundle {1,3}	yes	---	yes		25 - ?
Bundle {2,3}	---	yes	yes	<number>	

Our price prediction strategy is a modification of straightforward bidding that takes into account a vector of predicted prices (and I'll talk about how we come up with such predictions shortly).

Price Predictors

Estimate final prices:

- Perceived prices
- Vector of predicted prices
- Adjusted prediction
- Best bundle
 - maximizes surplus at adjusted predicted prices
 - assumes it will win the whole bundle

Slots	1	2	3	
Current prices (CP)	\$10	\$9	\$5	
Slots the agent is winning	yes	---	---	
Perceived prices	\$10	\$10	\$6	
Vector of predicted prices (π)	\$20	\$15	\$1	
Adjusted prediction = $\max\{\text{perc}, \pi\}$	\$20	\$15	\$6	
Job length	2 slots			
Deadline values	---	\$35	\$25	Surplus = Value - Cost
Bundle {1,2}	yes	yes	---	35 - 35 = \$0
Bundle {1,3}	yes	---	yes	25 - 26 = -\$1
Bundle {2,3}	---	yes	yes	25 - 21 = \$4

The idea is, instead of using the perceived prices, a price predicting bidder simply uses its price prediction to determine the optimal bundle of slots to bid on. Of course, if the price quote ever exceeds a prediction, like for slot 3 here, then the prediction was obviously wrong and is adjusted. In other words, a predicting agent is just like a SB except it uses as perceived prices the max of its predictions and its original perceived prices. And we see the difference it makes in this case: slot 1 is predicted to be very expensive so the most attractive bundle is now {2,3}.

Predictors

- SB = Straightforward Bidding = $PP(0)$
- BL = Baseline = Predict average prices for SB agents
- SC = Self-Confirming
- ECE = Expected Competitive Equilibrium
- EDCE = Competitive Equilibrium for expected demand



Prediction Methods	Predicted Final Price Vectors				
SB ($=PP(0)$)	0	0	0	0	0
$PP(\pi^{BL})$	14.8	10.7	7.6	4.6	1.9
$PP(\pi^{SC})$	13.0	8.7	5.4	3.0	1.2
$PP(\pi^{ECE})$	26.0	14.2	6.9	2.5	0.3
$PP(\pi^{EDCE})$	20.0	12.0	8.0	2.0	0.0

So that's how an agent turns price predictions into a bidding strategy. But how does it come up with price predictions? We considered 5 different price predictors.

The first is to predict all zeros which is not a predictor at all but just our old friend SB which is the same as predicting 0 prices because of this [back] max adjustment step.

Next, what we call our baseline prediction is to just predict the average prices achieved by all straightforward bidders, found by monte carlo simulation.

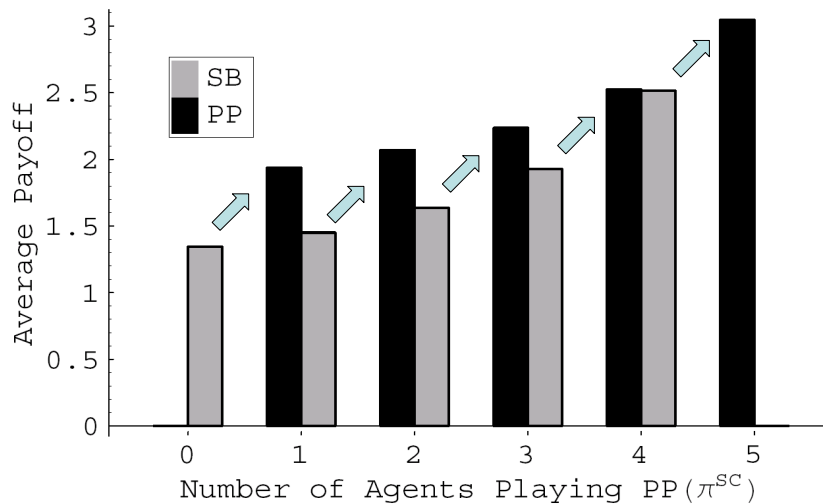
Self Confirming prediction is defined as a prediction vector such that if all agents use that prediction, the prediction will be true in expectation.

We found that by some extensive simulation [click]. Starting with an arbitrary price prediction we simulate a slough of games to find prices achieved by those price predictors, then start over with agents predicting *those* prices and keep iterating until we reach a fixed point.

Finally, we added two price predictions based on economic theory, which I won't describe here, but they're described in the paper.

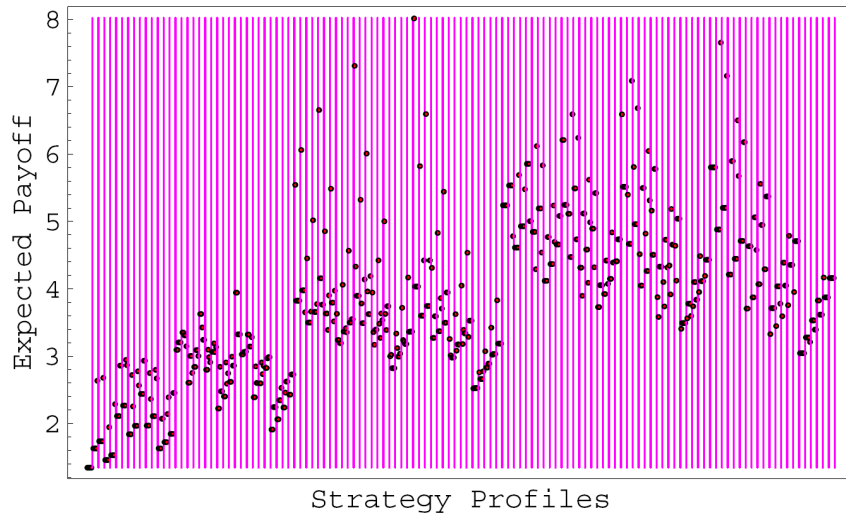
[click] And here are the resulting price vectors.

Example Payoff Matrix (2 strategies)



Now that we have some candidate strategies, how do they do? How any one agent performs depends on the strategies of the other agents, which means we need to create a payoff matrix. Here's a representation of a payoff matrix for a particular environment (5 agents, 5 slots, preferences drawn from a particular distribution) where the possible strategies available to any agent are just straightforward bidding vs price prediction with self confirming prices. It turns out that price prediction blows SB away in this case. In general, we want to find Nash Equilibria and we can find the only equilibrium (in fact, a dominant strategy) here by inspection. Notice that [click] if everyone is SB then you have an incentive to deviate to predicting. [click] and that holds for all profiles, leaving all predictors as the only profile no one would want to deviate from, and hence the only Nash Equilibrium.

Payoff Matrix for 5-Strategy Game



Of course, it's not always that easy. Here's a representation of the game where all 5 predictors are available strategies.

I won't try to parse this for you here, but in cases like this we need to employ various game solving tools to find equilibria.

It turns out that, unlike the last example, there's **no** profile of strategies here such that no one wants to deviate, which means the only Nash Equilibrium is in mixed strategies, meaning that your best strategy is to randomize between 2 or more other strategies. I'll say what those strategies are shortly...

Participation-Only Prediction

- Predictors always beat straightforward bidders
- Why does prediction help?
- Decompose behavior of predictor
 - Finding the best bundle
 - Deciding whether to bid on it
- Modified (PO) Predictor:
 - Pick best bundle as per SB
 - Only bid on it if positive surplus at *predicted* prices

<number>

So, I hate to spoil the punchline, but price prediction blows away straightforward bidding in all cases; so we wanted to test the other part of our hypothesis about **why** price prediction is so helpful.

We did that by decomposing the behavior of the price predictor into 2 decisions – finding the best bundle, and deciding whether to bid on it (which is really one decision since the empty bundle is one of the possible bundles). But as a way to find out what's going on with price prediction, we tested a modified version that ignores its predictions in deciding the best bundle but then holds back on bidding on that bundle if it predicts the prices will be too high. We call this a participation-only predictor.

Equilibria and Efficiency Results

- 98% of performance improvement is due to correct choice of participation or not
- SC and EDCE are supported in an equilibrium of 5-predictor game
- Prediction greatly improves agent (buyer) performance with a small efficiency loss (ie, hurts the seller)
 - Buyer surplus three times greater for equilibrium price predictors than all SB agents
 - Market efficiency (aggregate utility as fraction of optimal allocation) drops from 87% to 86% <number>

The basic conclusion of that is that nearly all the benefit of prediction is not in choosing better bundles but just refraining from bidding on whatever bundle you pick, if you can predict that it will get too expensive. In other words, avoiding the exposure problem.

As to what predictions work best: in the game where all 5 predictors (including the degenerate predictor SB) are available as strategies (that's this game [back 2]) we found a Nash equilibrium that consisted of everyone randomizing with close to equal probabilities between the self-confirming prices and one of the econ theory predictions.

More generally, our results show that prediction is a huge win for agents (about 3 times greater surplus) and this comes at a cost of a slightly less efficient market.

(for the non-economists, we find market efficiency by solving the centralized version of the scheduling problem – ie, find the optimal allocation of slots to agents knowing everyone's preferences – and compare the aggregate utility in that case with the aggregate utility achieved by the market)

The reason that the agents can all do better yet aggregate utility goes down is that the seller is the one to suffer because of the smarter buyers. The buyers avoid spurious purchases at great savings to themselves but at high cost to the seller. The net result is slightly less efficient allocations.

Conclusion: Price Prediction Helps in SAA

- Price prediction can significantly improve performance by reducing the exposure risk
- Performance depends on the quality of prediction and on how it is used
- Computational game theoretic approach to assessing strategies
- Results specific to particular scheduling domain
- But this is the best known strategic approach to bidding in any Simultaneous Ascending Auctions
- We expect price distribution predictors to perform much better...

<number>

To conclude, we've shown that price prediction greatly helps performance for bidding in simultaneous ascending auctions with complementarities and the reason it helps is by avoiding the exposure problem.

By comparing different predictors and ways of using prediction (for example, whether or not to use participation-only prediction) we found that performance depends on prediction quality and how it's used.

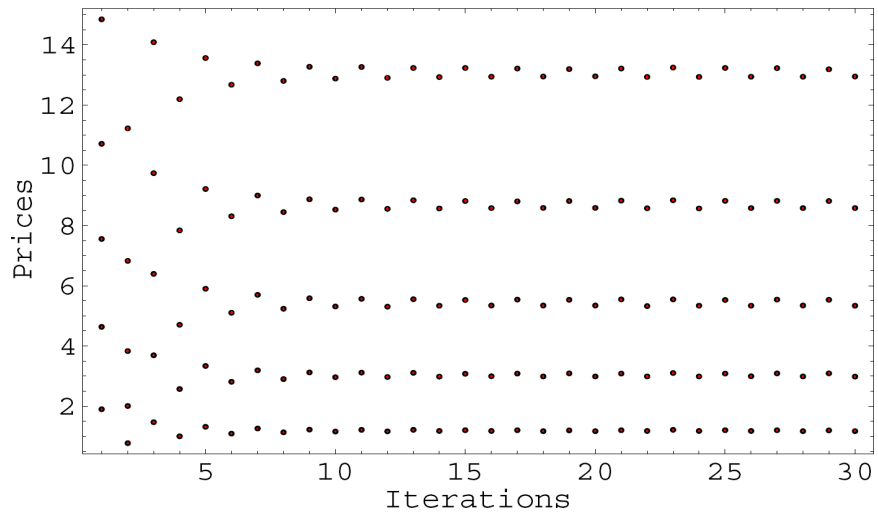
We've described our computational game theoretic approach of simulating games to derive empirical payoff matrices for strategy subsets and then finding Nash equilibria given the resulting payoff matrices.

As I've said, the results here are specific to a particular scheduling game but we expect the conclusions to apply generally to other Simultaneous Auction mechanisms.

Finally, we noted that performance depends on how a prediction is used. One way we'd expect to improve performance is to predict price *distributions* rather than single point predictions.

And in fact, that is being born out in our current work on this problem...

Convergence to Self-Confirming Price Predictions



Equilibria and Efficiency Results Table

- Click to add an outline

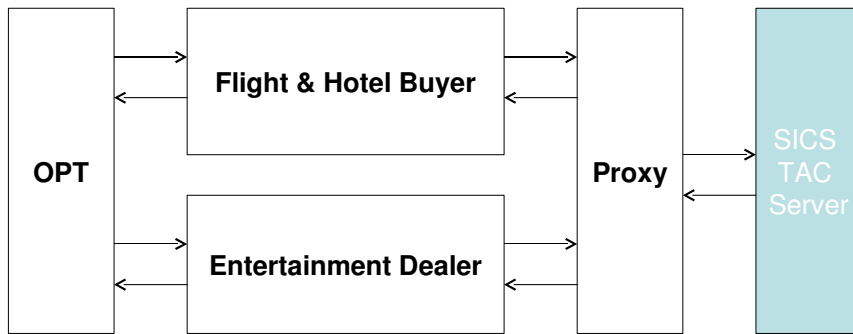
Games (i.e., strategy sets)	Equilibrium Profiles	% Eff.	Payoff	Average Final Price Vectors					
{SB, PP(π^{BL})}	all PP(π^{BL})	86	4.15	11.2	6.8	3.8	2.0	0.77	
{SB, PP(π^{BL}) w/ P.O.}	all PP(π^{BL}) w/ P.O.	85	4.07	11.8	6.9	3.7	1.7	0.58	
{SB, PP(π^{SC})}	all PP(π^{SC})	88	3.05	13.0	8.7	5.4	3.0	1.17	
{SB, PP(π^{BL}), PP(π^{SC}), PP(π^{ECE}), PP(π^{DCE})}	0.45 SC, 0.55 EDCE	86	4.25	10.6	6.5	4.0	2.2	0.91	
Additional Profiles									
	all SB	87	1.35	14.8	10.7	7.6	4.6	1.90	
	all PP(π^{ECE})	74	5.80	4.7	2.1	1.7	1.2	0.55	
	all PP(π^{DCE})	83	5.24	8.1	4.5	2.7	1.6	0.70	

Walverine

a TAC-02 Agent from the
University of Michigan



Architecture



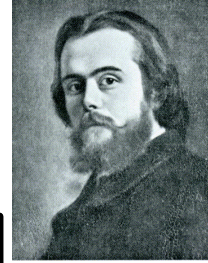
<number>

Flight & Hotel Loop

- Initial
 - Get flight prices
 - Initial predict
 - Client-by-client best-trip optimization
 - Buy flights
- Starting at 3:00, each minute:
 - Get quotes, transactions
 - Price prediction
 - Optimal package, buy flights if nec.
 - Get marginal values
 - Construct hotel bids

<number>

Price Prediction



Given initial flight prices, calculate
Walrasian competitive equilibrium
hotel prices.

Premises:

- Trip choices driven in large part by relative flight prices.
- Aggregate behavior reasonably approximated by competitive model.

<number>

Calculating Competitive Eq.

- Iterative price adjustment (tatonnement):

$$p_{t+1} \leftarrow p_t + \alpha_t [x(p_t) - 16]$$

where $x(p_t)$ is aggregate demand at hypothetical prices

- Demand estimation
 - For our own clients (8), calculate hotel rooms demanded for best package at hypothetical prices
 - For other agents' clients (56), employ [analytic expression for expected demand](#) based on client preference distribution
- Mid-game:
 - Employ quote/closing as price floors
 - Fix own demand at holdings for closed hotels

<number>

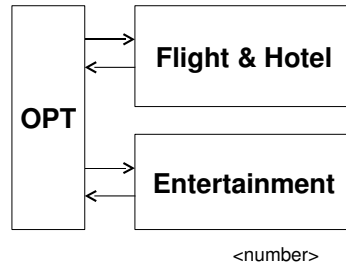
Hedging

- Equilibrium method yields point estimate, decisions highly sensitive.
- “Outlier probability”
 - Represents likelihood that prediction is wrong for a given hotel.
 - Outlier prediction defined as $\max(2p, 400)$.
 - Wolverine uses 0.06 per open hotel.
- Results in hedged package choices, hotel valuations.

<number>

Optimizer

- Integer linear program representing optimal package allocation
- Inputs: prices (actual, predicted), holdings
 - Reported separately by hotel, entertainment modules
- Outputs:
 - Optimal package
 - Marginal values (each unit)
 - Hedged marginal values



Hotel Bidding

- Compute hedged MV given predicted prices
- Compute *optimal shading*
 - Separately for each unit
 - Based on analytic model of other clients' MV distribution (similar to demand calc. approach)
 - Maximize bid value, accounting for:
 - Prob of winning unit & setting price
 - Prob of winning & not setting
 - Expected price if no bid
 - Number of other units affected
- Adjust optimal shades for BTQ rule

} conditional on ASK

<number>

Entertainment Dealing

- Derived a trading policy via Q-learning.
- Action is bid
 - to buy/sell unit in given entertainment auction
 - represented as offset from marginal value
- State space:
 - Game time, MV, holdings, Bid/Ask, day
 - Coarse distinctions, still 12852 states
- Rewards: Entertainment cash flow + fun bonus

<number>

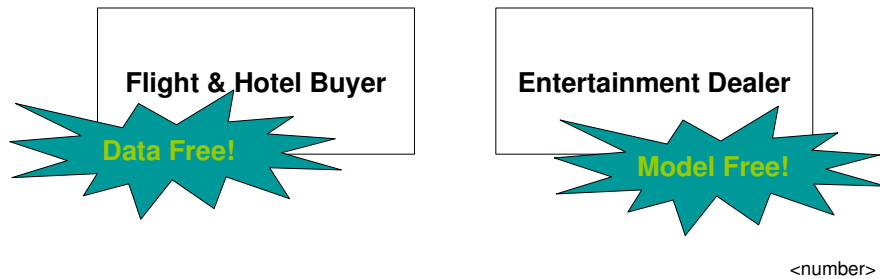
Learning to Trade

- Two giant Q-tables shared by all auctions (one each for days 1/4, 2/3)
- Played various policies, gathering transition and reward data
 - livingagents, “Exploit”, “Explore”
 - Games on SICS/own servers, variously populated
 - 14839 total games (× 12 auctions)

<number>

Summary: Walverine

- Price prediction based on competitive eq.
- Model-based optimal bidding.
- Q-Learned entertainment trading policy.



Online Auction Environments

- Auctions are efficient mechanisms for allocating resources
- Online auction space growing
 - Consumer – ebay, amazon
 - B2B
 - Electronic Trading Network (finance)
- Agents aid in the automation of trade in such auctions

<number>

Flights

- Agents are buyers
- One flight per day each way
- Prices determined by stochastic process
 - Random walk with upward drift

→ In expectation, flight prices increase over time, but different auctions increase at different rates

<number>

Hotels

- Two hotels (Towers, Shanties)
- Each with fixed number (16) of rooms available per day.
- Sold in simultaneous ascending 16th price auctions.
- To avoid sniping: One hotel closes randomly every minute

<number>

Entertainment Tickets

- Fixed pool of tickets for Museum, Amusement Park, Alligator Wrestling, by date, divided among agents.
- Clients have different preferences for event type.
- Agents trade among themselves through Continuous Double Auctions (e.g., stock market)

<number>

Agent Objectives

- Maximize total “profit”:
[sum over clients: trip utility] minus expenditures
- Client preferences: arrive/depart days, hotel premium, entertainment prefs
- Feasible trip: round trip airline, hotel room for interval
- Trip utility:
 - zero if infeasible
 - if feasible... (next)

<number>

Feasible Trip Utility

- Trip utility =
1000 – travel penalty + hotel bonus + fun bonus
- travel penalty = 100 per day deviation
- hotel bonus = {1 if Towers} × premium
- fun bonus =
Sum over types: {1 if ticket} × value

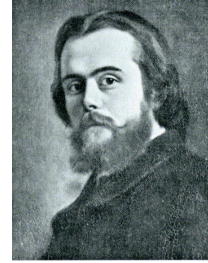
<number>

HotelAgent behavior

- First 30 seconds:
 - Generate hotel price predictions
 - Calculate optimal package, hedging for price volatility
 - Purchase flights in optimal package
- Before each hotel closing
 - Update price predictions
 - Calculate hedged marginal values of hotels
 - Calculate/submit an optimal bid for each hotel

<number>

Price Prediction



- Given
 - Distribution over client preferences
 - Assumptions about other agents' bidding
 - Known initial flight prices
- Compute Walrasian equilibrium prices for hotels
 - Prices for which supply meets expected demand

<number>

Hedging

- Equilibrium method yields point estimate, decisions highly sensitive.
- “Outlier probability”
 - Represents likelihood that prediction is wrong for a given hotel.
- Calculate hedged package choices, hotel valuations.

<number>

Hotel Bidding

- Compute hedged MV given predicted prices
- Compute *optimal bids*
 - Separately for each unit
 - Based on analytic model of other clients' MV distribution (similar to demand calc. approach)
 - Maximize bid value, accounting for:
 - Probability of winning
 - Expected impact on closing price

<number>

Walverine's Entertainment bidding:

- Entertainment accounts for ~40% of Walverine's total score.
- Our approach: Completely model-free.
- Policy derived through Q-learning algorithm.
- Reward: sum of cash-flow plus fun bonus.
- State space defined in terms of six dimensions: time, bid/ask quotes, ticket holdings, and marginal values to buy/sell.
- Actions: bid in terms of offsets from marginal value. <number>

Decision Tree for Flight Buying

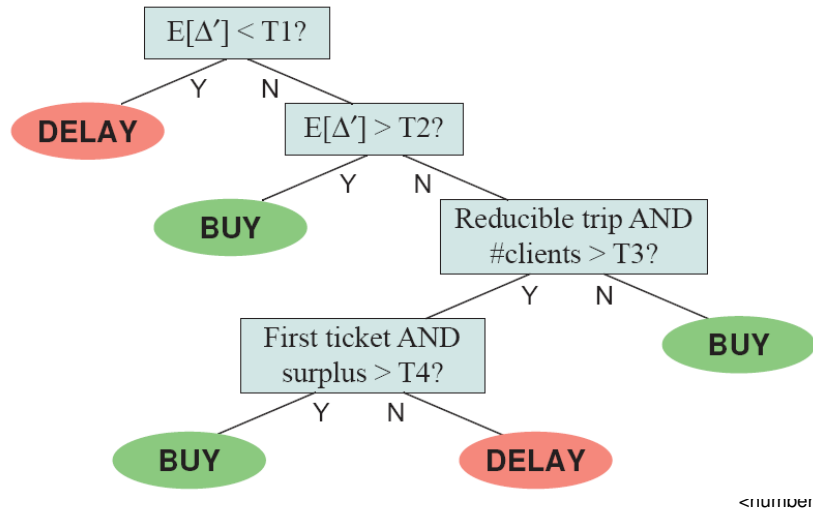


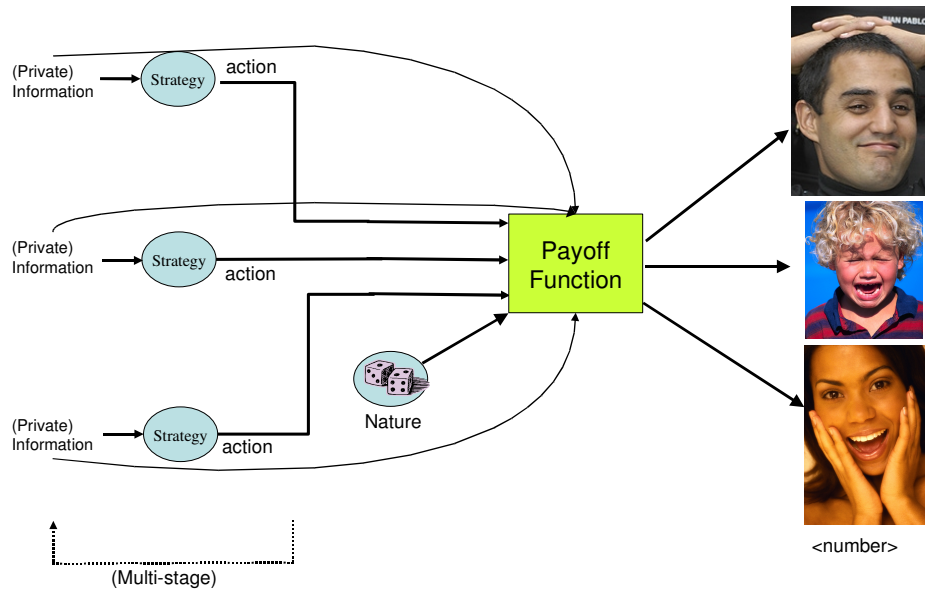
Figure 6.2: Decision tree for deciding whether to delay flight purchases.

How to Bid in Ebay

- Single good with independent private values: Bid your max!
- Multiple goods
 - Need to know the number of bidders and type distribution (a probability distribution over possible valuation functions for bundles of goods)
 - Compute self-confirming distribution prediction

<number>

Game Theory Primer



Actions, infinite game, payoff function, types, incomplete information, Nature, one-shot vs multi-stage game, strategy,

Randomness from Nature

