# New Approach

- Start with optimal allocator
  - No longer an advantage
  - Use as subroutine over many possible futures
- Learn model of expected price **distributions**
  - New boosting-based conditional density estimation algorithm for price prediction [ICML, 2002]
- Decision theoretic reasoning over predictions [AMEC, 2002]

# High-Level Strategy

- For each auction:
  - Repeatedly sample price vector from distributions
  - Bid avg marginal expected utility: $v(G_w^*) - v(G_l^*)$
- Bid for all goods — not just those in $G^*$

$\boxed{\textbf{Goal: analytically calculate optimal bids}}$

# Toy Example

| | utility |
|---|---|
| camera alone | \$50 |
| flash alone | 10 |
| both | 100 |
| neither | 0 |

- What's the value of the flash?
  - Camera price = \$70 $\Rightarrow$ value(flash) = \$30
  - Camera price = \$20 $\Rightarrow$ value(flash) = \$50
  - Camera price = \$40 $\Rightarrow$ value(flash) = \$50

- Expected value: resample camera price, take avg.

# Assumptions

1. Closing prices somewhat predictable
2. Own bids don't affect final prices
3. Can change bids in open auctions
4. Future decisions made with complete information

# Hotel Price Prediction

- Need hotel price predictions at various points in game
- Use machine learning methods applied to data from previously played games
- Want to estimate:
  - expected price
  - distribution of prices given current conditions (permits sampling of prices)

# Setting up the Learning Problem

- **Features:**
  - Current hotel and flight prices
  - Current time in game
  - Hotel closing times
  - Agents in the game (when known)
  - Variations of the above
- Built **8 separate predictors**:
  - Good Hotel vs. Bad (2)
  - Day 2,3 vs. day 1,4 — exploit symmetry (2)
  - 1st min. (pre-hotel-quotes) vs. later (2)
- **Data**: seeding round games
  - Features $\mapsto$ actual clear prices

# The Learning Algorithm

- $X$ = feature vector
- Given $X$, estimate distribution of $Y$ = ( hotel closing price ) − ( current bid )
- Break $Y$ into $k \approx 50$ cut points $b_1 \leq \cdots \leq b_k$
- For each $b_i$, estimate probability $Y \geq b_i$, given $X$
  - Can convert to estimated distribution of $Y|X$
- Say $X$ belongs to class $C_i$ if $Y \geq b_i$
  - $k$-class problem: each example in many classes
- Use **BoosTexter** (boosting “decision stumps”)
  - Can convert output to class prob. estimates

## The Learning Algorithm (cont.)

- Use logistic regression form of boosting
- Given $X$, get estimates $p_j$ of $\Pr[Y \geq b_j]$
- So can estimate $\Pr[b_j \leq Y < b_{j+1}]$ by $Q(j) = p_j - p_{j+1}$
- Gives distribution $Q$ over bins $[b_j, b_{j+1})$
- To sample:
  - Choose one bin at random according to $Q$
  - Choose point within bin uniformly at random
- Piecewise constant density function



Department of Computer Sciences
The University of Texas at Austin

# Hotel Expected Values

- Repeat until time bound, for each hotel:
  1. Assume this hotel closes next
  2. Sample prices from predicted price distributions
  3. Given these prices compute $V_0, V_1, \ldots V_8$
     - $V_i = v(G^*)$ if own **exactly** $i$ of the hotel
     - $V_0 \leq V_1 \leq \ldots \leq V_8$
- Value of $i$th copy is avg( $V_i - V_{i-1}$ )



Department of Computer Sciences
The University of Texas at Austin

# Other Uses of Sampling

**Flights:** Cost/benefit analysis for postponing commitment

**Cost:** Price expected to rise over next $n$ minutes

**Benefit:** More price info becomes known

- Compute expected marginal value of buying some different flight

**Entertainment:** Bid more (ask less) than expected value of having one more (fewer) ticket



Department of Computer Sciences
The University of Texas at Austin

# Finals

| Team | Avg. | Adj. | Institution |
|---|---|---|---|
| ATTac | 3622 | 4154 | AT&T |
| livingagents | 3670 | 4094 | Living Systems (Germ.) |
| whitebear | 3513 | 3931 | Cornell |
| Urlaub01 | 3421 | 3909 | Penn State |
| Retsina | 3352 | 3812 | CMU |
| CaiserSose | 3074 | 3766 | Essex (UK) |
| Southampton | 3253* | 3679 | Southampton (UK) |
| TacsMan | 2859 | 3338 | Stanford |

- ATTac improves over time
- livingagents is an open-loop strategy

# Controlled Experiments

- *ATTac*$_s$: "“full-strength” agent based on boosting
- *SimpleMean*$_s$: sample from empirical distribution (previously played games)
- *ConditionalMean*$_s$: condition on closing time
- *ATTac*$_{ns}$, *ConditionalMean*$_{ns}$, *SimpleMean*$_{ns}$: predict expected value of the distribution
- *CurrentPrice*: predict no change
- *EarlyBidder*: motivated by TAC-01 entry livingagents
  - Immediately bids high for $G^*$ (with *SimpleMean*$_{ns}$)
  - Goes to sleep

# Stability

- 7 *EarlyBidder*'s with 1 *ATTac*

| Agent | Score | Utility |
|---|---|---|
| *ATTac* | $2431 \pm 464$ | $8909 \pm 264$ |
| *EarlyBidder* | $-4880 \pm 337$ | $9870 \pm 34$ |

- 7 *ATTac*'s with 1 *EarlyBidder*

| Agent | Score | Utility |
|---|---|---|
| *ATTac* | $2578 \pm 25$ | $9650 \pm 21$ |
| *EarlyBidder* | $2869 \pm 69$ | $10079 \pm 55$ |

*EarlyBidder* gets more utility; *ATTac* pays less


Department of Computer Sciences
The University of Texas at Austin

# Results

- *Phase I*: Training from TAC-01 (seeding round, finals)
- *Phase II*: Training from TAC-01, phases I, II
- *Phase III*: Training from phases I – III

| *Agent* | *Relative Score* | |
|---|---|---|
| | *Phase I* | *Phase III* |
| $ATTac_{ns}$ | $105.2 \pm 49.5$ (2) | $166.2 \pm 20.8$ (1) |
| $ATTac_{s}$ | $27.8 \pm 42.1$ (3) | $122.3 \pm 19.4$ (2) |
| *EarlyBidder* | $140.3 \pm 38.6$ (1) | $117.0 \pm 18.0$ (3) |
| $SimpleMean_{ns}$ | $-28.8 \pm 45.1$ (5) | $-11.5 \pm 21.7$ (4) |
| $SimpleMean_{s}$ | $-72.0 \pm 47.5$ (7) | $-44.1 \pm 18.2$ (5) |
| $ConditionalMean_{ns}$ | $8.6 \pm 41.2$ (4) | $-60.1 \pm 19.7$ (6) |
| $ConditionalMean_{s}$ | $-147.5 \pm 35.6$ (8) | $-91.1 \pm 17.6$ (7) |
| *CurrentPrice* | $-33.7 \pm 52.4$ (6) | $-198.8 \pm 26.0$ (8) |