Motivation

- Classic environment properties of MAS
  - Stochastic behavior (agents and environment)
  - Incomplete information
  - Uncertainty

- Application Examples
  - Robotics
  - Intelligent user interfaces
  - Decision support systems
Motivation

- Popular environment: Texas Hold’em poker
  - Enjoyed by users
  - Interaction with agents
  - Many solutions

- Annual Computer Poker Challenge (ACPC)
  - Held with AAAI conference
  - Existing game framework
  - Competition!
Overview

- Background
- Methodology
- Results
- Conclusions
Texas Hold’em Poker

- Variant of poker developed in Robstown, Texas in early 1900s
- Played with 52 card deck
Background | Texas Hold’em Poker

- Ranking of poker hands

![Diagram of poker hands](http://www.learn-texas-holdem.com/)

Texas Hold’em Poker

- Uses both 2 private and 5 community cards
- Construct the best possible poker hand out of 5 cards (use 3-5 community)

Background | Methodology | Results | Conclusions
---|---|---|---

**private cards**

![Cards](image1)

**community cards**

![Cards](image2)

(best poker hand)
Games consist of 4 different steps

- Actions: bet (check, raise, call) and fold
  - Bets can be limited or unlimited

Background Methodology Results Conclusions

(pre-flop) private cards community cards

(1) pre-flop

(2) flop

(3) turn

(4) river
Significant worldwide popularity and revenue
- World Series of Poker (WSOP) attracted 63,706 players in 2010 (WSOP, 2010)
- Online sites generated estimated $20 billion in 2007 (Economist, 2007)

Has fortuitous mix of strategy and luck
- Community cards allow for more accurate modeling
- Still many “outs” or remaining community cards which defeat strong hands
Strategy depends on **hand strength** which changes from step to step!

- Hands which were strong early in the game may get weaker (and vice-versa) as cards are dealt.
Strategy also depends on **betting behavior**

Three different types (Smith, 2009):
- Aggressive players who often bet/raise to force folds
- Optimistic players who often call to stay in hands
- Conservative or “tight” players who often fold unless they have really strong hands
Problem: provide basic strategies that simulate betting behavior types
- Must include hand strength
- Must incorporate stochastic variance or “gut feelings”
  - Action: fold/call with high/low hand strength
Solution 1: use separate mixture models for each type

- All three models use the same set of three tactics for weak, medium, and strong hands
  - Each tactic uses a different probability distribution for actions (raise, check, fold)
- However, each model has a different idea what hand strength constitutes a weak, medium, and strong hand!
Solution 2: Probability distributions

Hand strength measured using Poker Prophesier (http://www.javaflair.com/pp/)

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Weak</th>
<th>Medium</th>
<th>Strong</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggressive</td>
<td>[0…0.2)</td>
<td>[0.2…0.6)</td>
<td>[0.6…1)</td>
</tr>
<tr>
<td>Optimistic</td>
<td>[0…0.5)</td>
<td>[0.5…0.9)</td>
<td>[0.9…1)</td>
</tr>
<tr>
<td>Conservative</td>
<td>[0…0.3)</td>
<td>[0.3…0.8)</td>
<td>[0.8…1)</td>
</tr>
</tbody>
</table>

(1) Check hand strength for tactic

(2) “Roll” on tactic for action

<table>
<thead>
<tr>
<th>Tactic</th>
<th>Fold</th>
<th>Call</th>
<th>Raise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weak</td>
<td>[0…0.7)</td>
<td>[0.7…0.95)</td>
<td>[0.95…1)</td>
</tr>
<tr>
<td>Medium</td>
<td>[0…0.3)</td>
<td>[0.3…0.7)</td>
<td>[0.7…1)</td>
</tr>
<tr>
<td>Strong</td>
<td>[0…0.05)</td>
<td>[0.05…0.3)</td>
<td>[0.3…1)</td>
</tr>
</tbody>
</table>
Methodology | Meta-strategies

- Problem: basic strategies are very simplistic
  - Little emphasis on **deception**
  - Don’t **adapt** to opponent

- Consider four meta-strategies
  - Two as baselines
  - Two as active AI research
Methodology | Deceptive Agent

- Problem 1: Agents don’t explicitly **deceive**
  - Reveal strategy every action
  - Easy to model

- Solution: alternate strategies periodically
  - Conservative to aggressive and vice-versa
  - Break opponent modeling (concept shift)
Problem 2: Basic agents don’t adapt
- Ignore opponent behavior
- Static strategies

Solution: use reinforcement learning (RL)
- Implicitly model opponents
- Revise action probabilities
- Explore space of strategies, then exploit success
RL formulation of poker problem

- **State s**: hand strength
  - Discretized into 10 values

- **Action a**: betting behavior
  - Fold, Call, Raise

- **Reward R(s,a)**: change in bankroll
  - Updated after each hand
  - Assigns same reward to all actions in a hand
Methodology | Explore/Exploit

- Q-Learning algorithm
  - Discounted learning
  - Single-step only

- Explore/Exploit balance
  - Choose actions based on expected reward
  - Softmax
    - Probabilistic matching strategy
    - Used by humans (Daw et. al, 2006)
    - Roulette selection
      \[
      P(\alpha | s) = \frac{e^{R(s,\alpha)}}{\sum_{\alpha' \in A} e^{R(s,\alpha')}}
      \]
Methodology | Active Sensing

- Opponent modeling
  - Another approach to adaptation
  - Want to understand and predict opponent’s actions
  - **Explicit** rather than implicit (RL)

- Primary focus of previous work on AI poker
  - Not proposing a new modeling technique
    - Adapt existing techniques to basic agent design
  - Vehicle for fundamental agent research
Methodology | Active Sensing

- Opponent model = knowledge
  - Refined through observations
    - Betting history, opponent’s cards
  - Actions produce observations
    - Information is not free

- Tradeoff in action selection
  - Current vs. future hand winnings/losses
  - Sacrifice vs. gain
Methodology | Active Sensing

- Knowledge representation
  - Set of Dirichlet probability distributions
    - Frequency counting approach
    - Opponent state $s^o = \text{their estimated hand strength}$
    - Observed opponent action $a^o$
      \[
      P(a|s^o) = \frac{c(s^o,a^o)}{\sum_{a^{o'} \in A} c(s^o,a^{o'})}
      \]
  - Opponent state
    - Calculated at end of hand (if cards revealed)
    - Otherwise $1 - s$
      - Considers all possible opponent hands
Methodology | Active Sensing

- Challenge: how to choose actions?
  - Goal 1: Win current hand
  - Goal 2: Win future hands (good modeling)
  - Goals can be conflicting

- Another exploration/exploitation problem!
  - Explore: learn opponent model
  - Exploit: use model in current hand
Exploitation

Use opponent actions to revise hand strength model

- Have $P(a^o | s^o)$
- Estimate $P(s^o | a^o)$
- Use Bayes rule
  
  $$P(s^o | a^o) = \frac{P(s^o | a^o) P(a^o)}{P(s^o)}$$

Action selection

- Raise if our hand strength $>> E[P(s^o | a^o)]$
- Call if our hand strength $\approx E[P(s^o | a^o)]$
- Fold if our hand strength $<< E[P(s^o | a^o)]$
Methodology | Active Sensing

- Use adaptive $\epsilon$-greedy approach
  - Explore with probability $w \times \epsilon$
  - Exploit with probability $1 - w \times \epsilon$

- Control adaptive exploration through $w$
  - $w = \text{entropy of } P(s^o \mid a^o)$
  - High when probabilities most similar
    - High uncertainty
  - Low when probabilities diverse
    - Low uncertainty

Background | Methodology | Results | Conclusions
Methodology | Active Sensing

1. Opponent Model
2. Analyze Opponent Model
3. Compute Entropy
4. Choose Explore Action
5. Explore Exploit
6. Choose Exploit Action

- \( P(a^o | s^o) \)
- \( P(s^o | a^o) \)
- \( c(s^o, a^o) \)

Agent

Observations
Problem 1: Current strategies (basic and EE) focus only on hand strength
- No thought given to other “features” such as betting sequence, pot odds, etc.
- No thought given to previous hands against same opponent

Such a myopic approach limits the reasoning capability for such agents

Solution 1: Strategy should consider entire “session” including all the above features
Problem 2: Different strategies may only be effective against certain opponents

- Example: Doyle Brunson has won 2 WSOP with 7-2 off suit—worst possible starting hand
- Example: An aggressive strategy is detrimental when opponent knows you are aggressive

Solution 2: Choose the “correct” strategy based on the previous sessions
Approach 2: Find the Boundary of Use (BoU) for the strategies based on previously collected sessions

- BoU partitions sessions into three types of regions (successful, unsuccessful, mixed) based on the session outcome

- Session outcome—complex and independent of strategy

- Choose the correct strategy for new hands based on region membership
BoU Example

<table>
<thead>
<tr>
<th>Boundary of Use</th>
<th>Unsuccessful</th>
<th>Successful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Border</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Strategy Incorrect
- Strategy Correct

Ideal: All sessions inside the BoU
Approach 2. Improve the BoU using focused refinement (on mixed regions)

- Repair session data to make it more beneficial for choosing the strategy
  - Active learning
  - Feature selection
- Update the strategies chosen (based on the “repaired” sessions) which may change outcome
Methodology | BoU

- BoU Framework

Based on previous poker sessions

Create the BoU → Regions With Sessions → Apply Session Repair

 Sessions With Outcomes → Start → Steps 1-2 (E) → Repaired Sessions

Yes: More Successful? → Sessions with New Outcomes → Update Model → Step 3 (M)

No: Step 4

Using query synthesis and feature selection

For the basic strategies

End
Challenges (to be addressed)

- How do we determine numeric outcomes?
  - Amount won/lost per hand
  - Correct action taken for each step

- How do we assign region types to numeric outcomes?
  - Should a session with +120 outcome and a session with +10 both be in successful region?

- How do we update outcomes using the strategies?
  - Say we switch from conservative to aggressive so the agent would not have folded
  - How do we simulate the rest of the hand to get the session outcome?
BoU Implementation

- **k-Means clustering**
  - Similarity metric needs to be modified to incorporate *action sequences* AND *missing values*
  - Number of clusters used must balance cluster purity and coverage

- **Session repair**
  - Genetic search for subsets of features contributing the most to session outcome
  - Query synthesis for *additional hands* in mixed regions
Results | Overview

- Validation
  - Basic agent vs. other basic (DONE)
  - EE agent vs. basic agents (DONE)
  - Deceptive agent vs. EE agent

- Investigation
  - AS agent vs. EE/deceptive agents
  - BoU agent vs. EE/deceptive agents
  - AS agent vs. BoU agent
    - Ultimate showdown
Simple Agent Validation

- Simple Agent Hypotheses
  - SA-H1: None of these strategies will “dominate” all the others
  - SA-H2: Stochastic variance will allow an agent to win overall against another with the same strategy

- Parameters
  - Hands = 500
  - Seeds = 30
Results | Simple Agent Validation

- Matchups
  - Conservative vs. Aggressive (DONE)
  - Aggressive vs. Optimistic (DONE)
  - Optimistic vs. Conservative (DONE)
  - Aggressive vs. Aggressive (DONE)
  - Optimistic vs. Optimistic (DONE)
  - Conservative vs. Conservative (DONE)
Matchup 1: Conservative vs. Aggressive
Results | Simple Agent Validation

- Matchup 2: Aggressive vs. Optimistic

![Graph showing Aggressive vs. Optimistic Winnings](image)

- Aggressive Winnings vs. Round Number
- Won/Lost
Matchup 3: Optimistic vs. Conservative

Optimistic vs. Conservative

Optimistic Winnings vs. Round Number

Graph showing the comparison of Optimistic and Conservative winnings over rounds.
Results | EE Validation

- EE Hypotheses
  - EE-H1: Explore/exploit will lose money early while it is exploring
  - EE-H2: Explore/exploit will eventually adapt and choose actions which exploit simple agents to improve its overall winnings

- Parameters
  - Hands = 500
  - Seeds = 30
  - Learning Rate = Discounted
Matchup 1: EE vs. Aggressive

EE vs. Aggressive

- EE Winnings
- Round Number
- Won/Lost
Matchup 2: EE vs. Optimistic

EE vs. Optimistic

![Graph showing EE vs. Optimistic winnings over round number.]
Matchup 3: EE vs. Conservative

EE vs. Conservative

-200
0
200
400
600
800
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800

EE Winnings

Round Number

Won/Lost
Matchup 4: EE vs. Deceptive
Active Sensing Hypotheses

- AS-H1: Including opponent modeling will improve agent winnings
- AS-H2: Using AS to boost opponent modeling will improve agent winnings over non-AS opponent modeling

Open questions:

- How is agent performance affected by:
  - ε values?
  - Other opponent performs modeling?
Results | AS Setup

- Parameters
  - $\varepsilon = 0.0, 0.1, 0.2$

- Opponents
  - EE: implicit vs. explicit modeling, dynamic opponent
  - Deceptive: shifting opponent
  - Non-AS: effect of opponent’s modeling
  - BOU: Offline learning/modeling
Results | BoU Setup

- BoU Hypotheses
  - BoU-H1: Including additional session information should improve agent reasoning
  - BoU-H2: Using the BoU to choose the correct strategy should improve winnings over agents which only use hand strength

- BoU Data Collection
  - Simple agent validation
  - Crowdsourcing agents vs. humans
Conclusion | Remaining Work

- Finish implementing AS
- Finish implementing BOU
- Run AS/BOU Experiments
- POJI results
Introduction poker as an AI problem

Described various agent strategies
  - Basic
  - Need for meta-strategies
  - AS/BOU

Introduced experimental setup
  - Early validation results
Questions?
Demonstration
References

Acknowledgements

- Playing card images from David Bellot: http://www.eludication.org/playingcards.html#